DESIGN AND ANALYSIS OF EVERGREEN VIRTUALLY CLUSTERED AUTOMATION PLATFORM

BY

GHALIB A. ALHASHIM

A Dissertation Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

In

COMPUTER SCIENCE AND ENGINEERING

MAY 2016

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **GHALIB A. ALHASHIM** under the direction his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE AND ENGINEERING.**

Sadiq Sait. M

Dr. Sadiq M. Sait
(Advisor)

[illegible]

Dr. Adel F. Ahmed
Department Chairman

R E Abdel-Aal

Dr. Radwan Abdel-Aal
(Member)

[illegible]

Dr. Salam A. Zummo
Dean of Graduate Studies

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

Tarek Hel

Dr. Tarek H. El-Basuny
(Member)

24/5/16

Date

[illegible]

Dr. Fouad M. Al-Sunni
(Member)

[illegible]

Dr. Moataz Ahmed
(Member)

Dedicated to my parents and my family.

ACKNOWLEDGMENTS

I thank Allah, the Lord of worlds, for His mercy and limitless help and guidance. May peace and blessings be upon Mohammed the last of the messengers.

Acknowledgment is due to KFUPM for the support of this dissertation.

I would like express my deep appreciation to my advisor Dr. Sadiq M. Sait for his invaluable advice and encouragement. I also wish to thank the other members of my Dissertation committee Dr. Radwan Abdel-Aal, Dr. Tarek H. El-Basuny, Dr. Fouad M. Al-Sunni, and Dr. Moataz Ahmed for their helpful suggestions and comments.

I am grateful to my family, especially my wife and children for their patience and understanding during my busy schedule.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CCR	:	Centralized Control Room
CCS	:	Compressor Control System
CMS	:	Condition Monitoring System
COTS	:	Commercial Off-The-Shelf
CPU	:	Central Processing Unit
DCS	:	Distributed Control System
DDS	:	Data Distribution Service
EPS	:	Equipment Protection System
ESD	:	Emergency Shutdown System
F&GDS	:	Fire and Gas Detection System
GOSP	:	Gas-Oil Separation Plant
HMI	:	Human Machine Interface
I/O	:	Input/Output
IEC	:	International Electrotechnical Commission
JB	:	Junction Box
LAN	:	Local Area Network
Mbps	:	Megabits Per Second
OMG	:	The Object Management Group
PAS	:	Process Automation System
PC	:	Personal Computer
PIB	:	Process Interface Building
PLC	:	Programmable Logic Controller
QoS	:	Quality of Service
RPC	:	Remote Procedure Call
SCADA	:	Supervisory Control And Data Acquisition System
TMS	:	Terminal Management System
vCAP	:	Virtually Clustered Automation Platform

ABSTRACT

Full Name	:	Ghalib A. Al-Hashim
Thesis Title	:	Design And Analysis Of Evergreen Virtually Clustered Automation Platform
Major Field	:	Computer Science And Engineering
Date of Degree	:	May 2016

A paradigm shift from the traditional distributed control system architecture is proposed to address the obsolescence challenges resulting from the life cycle management of proprietary automation systems. It is based on ubiquitous data-centric architecture using distributed autonomous process interface systems, fault-tolerant and real-time data distribution service middleware, and virtually clustered automation and control servers. Many distributed control system architectures are proposed in the literature; however, none of them is fully based on standard software and hardware technologies. The proposed architecture relies on completely standard computing and networking technologies that can provide flexibility and heterogeneous scalability across multiple vendors. As a result, this new automation architecture will avoid any potential obsolescence challenges and will result in reducing the total cost of ownership from 30% to 66% throughout the life span of the processing facility. In addition, the initial capital cost of grass root process automation investment can be reduced by 20% by eliminating the requirements for environmentally controlled process interface buildings, systems and marshaling cabinets, and wire trays and cablings from junction boxes to process interface buildings.

ملخص الرسالة

الاسم الكامل: غالب عبدالرضى علي الهاشم

عنوان الرسالة: تصميم و تحليل منصة تحكم افتراضيه قابلة للتجديد المستمر

التخصص: علوم و هندسة الحاسب الآلي

تاريخ الدرجة العلمية: ديسمبر 2016

في هذه الرساله، نقدم نقلة نوعيه من الهيكل التقليدي في تصميم نظم التحكم الموزعه التي تُنتَج من قبل شركات متخصصه و مغلقه حيث لا تَقبل الإفصاح لغيرها عن طريقة تصنيع أي من قطع الغياراللازمه لإستدامة تلك الأنظمه مما يُؤدي الى الإضطرار لتجديدها بشكل مستمر و بتكلفة عالية جداً عند عدم توفر قطع الغيار لأي جزء منها. هذا الطرح الجديد يعتمد على هيكلة نظام التحكم بناءً على نقل المعلومات بشكل سلس و انسيابي و فعال من أيّ مصدر كان الى أيّ جزء من النظام الذي بحاجه الى تلك المعلومات. النظام يتكون من أنظمة مستقله و موزعه في معمل الإنتاج لتحويل الإشارات الكهربيه من أجهزة القياس و حساسات التصنيع الى معلومات رقميه، ومن برمجيات وسيطه لتوزيع المعلومات المخصصه لانظمة الوقت الحقيقي و المقيده بزمن استجابة معينه و القادرة على تحمل أي خلل في النظام ، و كذلك من خوادم التحكم و التشغيل الآلي المدمجه افتراضياً. لقد قُدّمت بحوث كثيرة في تصميم نظم التحكم الموزعه و لكن لا يوجد أي منها يعتمد على التكنلوجيا النموذجية للأجهزة والبرمجيات. على عكس ذلك، الهيكل المقدم في هذا الطرح يعتمد كليا على الحاسبات الآليه و شبكات الإتصال النموذجية التي بإمكانها توفير المرونه و التوافق اللازم لإستخدام قطع الغيار من عدة مسوقين بدون الإعتماد الكلي على أيّ مصنّع. كنتيجة لاستخدام هذا الهيكل الجديد للتحكم، من الممكن تقليل التكلفة الكلية بنسبة 30% الى 66% لاستدامة نظم التحكم على امتداد العمر الافتراضي للمعمل و معدات التصنيع. كذلك سوف تقلل تكلفة نظم تحكم التصنيع الجديدة بنسبة لا تقل عن 20% و ذلك للاستغناء عن بناء المباني المكلفة واللازمة لاحتواء اجهزة النظم التقليديه و كذلك مخازن تنظيم الاسلاك الكهربائيه ومخازن الاجهزة الالكترونية و تمديد الاسلاك اللازمة من مخازن التوصيل بالقرب من معدات التصنيع الى مخازن تنظيم الاسلاك الكهربائيه في هذه المباني.

CHAPTER 1

INTRODUCTION

Processing facilities in oil and gas, petrochemical, pharmaceutical, water/wastewater treatment, manufacturing, aerospace, travel, food, textile, film, hospital, leisure, foundry, agriculture, plastic and printing industries utilize various process automation and safety systems including distributed control system (DCS), supervisory control and data acquisition system (SCADA), terminal management system (TMS), compressor control system (CCS), equipment protection system (EPS), emergency shutdown system (ESD), condition monitoring system (CMS), fire and gas detection system (F&GDS), etc.., in order to operate the plants safely and reliably to the optimal and maximum extent possible. Each system is manufactured by a specific vendor utilizing specialized proprietary components. The level of operation optimality and efficiency depends on the availability of timely dynamic process data from all systems especially during abnormal situations. The overall system performance in furnishing dynamic process data updates depends highly on the level of integration among the heterogeneous process automation systems and the number of different manufacturing vendors making up the total process automation solution. Normally, the process automation system consists of a main DCS and a number of auxiliary sub-systems. DCS refers to a control system designed by one system manufacturer to monitor and control processing facilities in which the controller elements are not centralized in one location but are distributed throughout the plant system with each sub-system unit controlled by one or more controllers. Auxiliary sub-systems include ESD to act as a safety protection layer, CMS to monitor machinery vibration and temperature, CCS to control the performance of the rotating equipment, F&GDS to detect fire and dangerous gas leaks, and EPS to provide interlock for preventing machines from harming their operators or damaging themselves. The entire system of controllers is connected by networks for communication and monitoring. Figure 1 shows typical functional levels of a process automation system for controlling a gas oil separation plant (GOSP) used in the oil production industry.

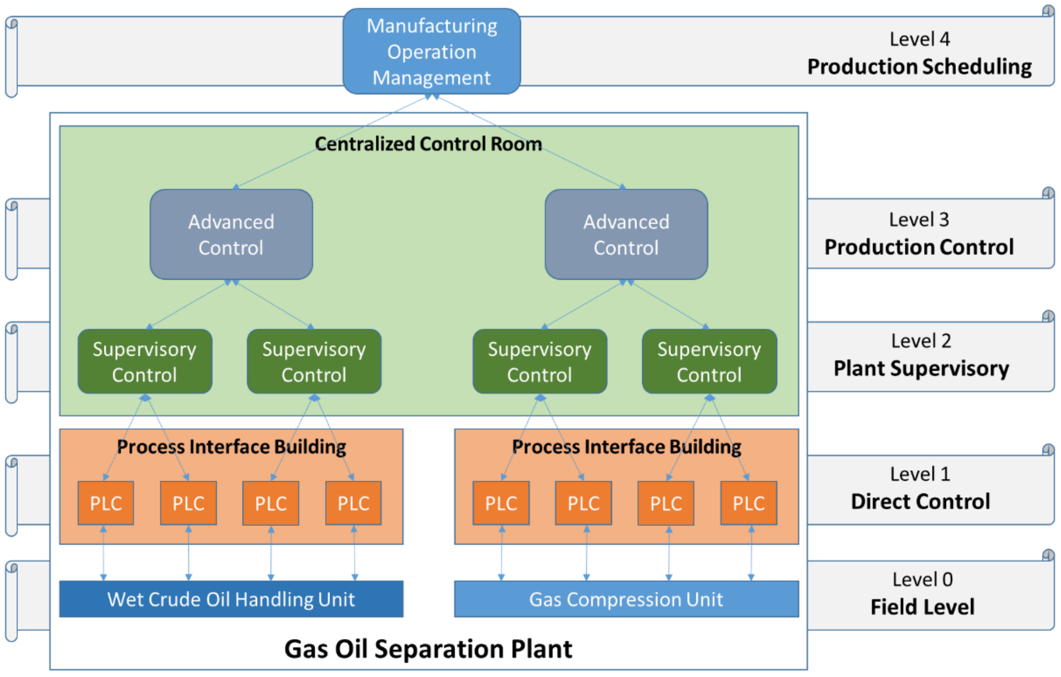


Figure 1 - Typical Functional Levels of a Process Automation System

The GOSP separates the crude oil from sediments, solids, sand, gases and condensates to allow the crude to be pumped on the pipeline. It receives untreated crude oil from production oil wells. The untreated crude oil contains saltwater and gas. This is known as wet crude oil. The processes in the GOSP separate gas and saltwater out from the untreated crude oil. The remaining crude oil and the gas are transported for further processing. The saltwater can be pumped back into the oil wells to help recover more oil. These plants are mostly located at the main oil fields. A GOSP can be located onshore or offshore. Larger oil fields may have more than one GOSP to process the untreated crude oil. [27]

A typical processing plant includes one centralized control room (CCR) and multiple process interface buildings (PIBs) distributed throughout the field to minimize the length of required cabling and wiring. In a typical GOSP, there is one CCR and two PIBs; one for the wet crude oil handling unit and one for the gas compression unit as shown in Figure 2. [77]

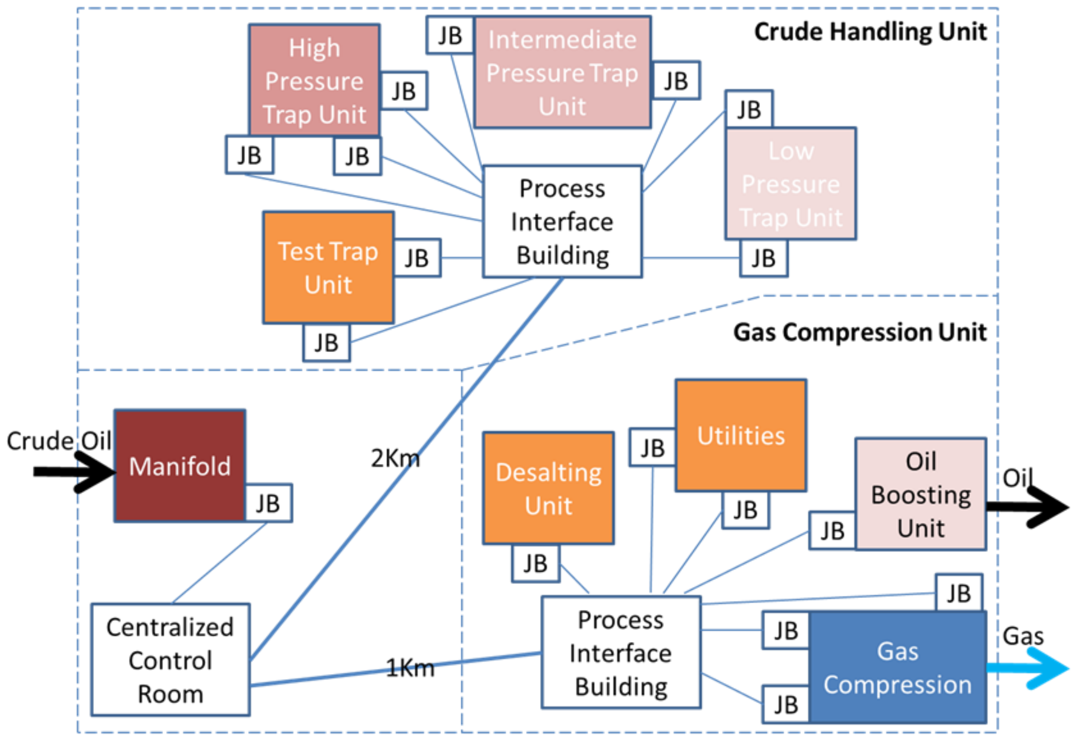


Figure 2 - Typical Layout of a GOSP

The field level in Figure 1 includes the process measurement sensors such as pressure, flow, level, and temperature instruments and final device elements such motor operated gate valves and control valves to regulate the process pressure, flow, level or temperature. All field instruments are hardwired to field junction boxes (JBs) close to the associated process equipment. Most of the field junction boxes are hardwired to the marshaling cabinets inside the process interface building closest to the process equipment. The remaining field junction boxes are hardwired all the way to the marshaling cabinets inside the rack room of the CCR as shown in Figure 2. The direct control level in Figure 1 includes various process automation and safety controllers enclosed in system cabinets located in the associated PIBs and CCR rack room. The plant supervisory level in Figure 1 includes various operation consoles including human machine interface (HMI) consoles and engineering and maintenance consoles. The production control level in Figure 1 includes advanced process control applications for overall plant control optimization. The production scheduling level in Figure 1 includes linear-programming models for optimum production planning and scheduling. [65] Most of the process automation systems (PAS) used in control applications are designed based on programmable logic controllers (PLCs).

1.1 PLC System Description

A PLC is a “hard” real-time digital computer used for automation of typically industrial electromechanical processes. It is a “hard” real-time system since the output results must be produced in response to any changes in input conditions within a limited time, otherwise unintended operation will result. It uses a programmable memory for storing process control instructions including logic sequencing, timing, counting, and arithmetic to control various types of equipment or processing facilities through process interface input/output (I/O) modules. This type of control system was invented in the 1970s and has made a significant contribution to the automation of manufacturing facilities. Earlier automation systems had to use thousands of hard wired relays and timers, which had to be rewired or replaced whenever there was a need to change the manufacturing models. [21]

A PLC consists of the following components illustrated in Figure 3: central processing unit (CPU), memory, input modules, output modules, and power supply. In the diagram, the green arrows indicate the power supply lines; and the blue and red arrows between blocks indicate the information flowing directions. The PLC is manufactured in two packaging styles, fixed and modular. On one hand, the fixed style is usually small, has less memory, and a limited number of input and output channels. The CPU, power supply, and I/O channels are all packaged in a single unit. A complete replacement of the unit would be required should any critical parts fail. This PLC type is relatively inexpensive and generally used for basic functions and not suited for future expansion. [74]

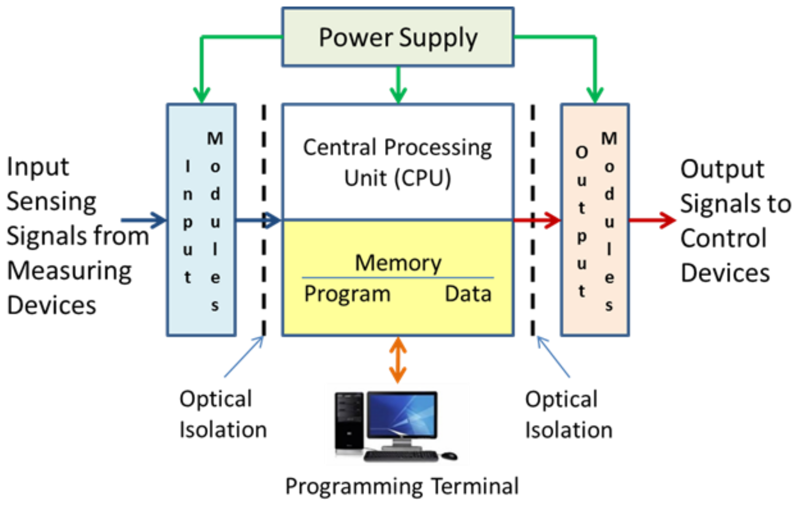


Figure 3 - PLC Hardware Block Diagram

On the other hand, the modular style, also referred to as rack-mounted PLC, has separate constructions for the CPU, power supply, I/O system, and expansion areas. It is based on a chassis, rack or base plate that allows for the installation of the main CPU module, power supply module, and numerous I/O modules or printed circuit boards contained inside a casing. The power supply module provides the required power along the backplane bus bar to the CPU and various I/O modules. Also, the CPU module communicates to the I/O modules along the backplane communication bus.

The I/O system contains pluggable sections so that modules can be mixed and matched. While this system tends to be expensive, the advantages outweigh the cost. It allows for future upgrades, expansion and increased application options, which prove beneficial economically. Moreover, modular PLCs offer simpler troubleshooting, ultimately alleviating system downtime.

There are four basic steps in the operation of all PLCs; input scan, program scan, output scan, and housekeeping. These steps form one composite scan time continually take place in a repeating 100ms-resolution loop as shown in Figure 4. During the input scan, the PLC input modules detect the state of all input sensors connected to the PLC. During the program scan, the PLC CPU executes the user created program logic. During the output scan, the PLC output modules energize or de-energize all final devices connected to the PLC. During the housekeeping step, the PLC CPU performs internal diagnostics and communications with programming terminals as well as human machine interface workstations. [74]

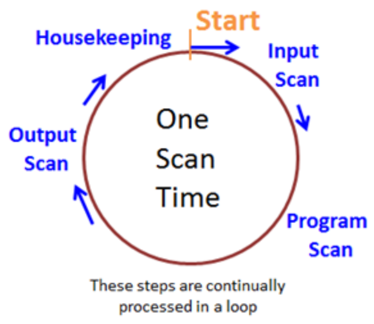


Figure 4 - PLC Operation

The International Electrotechnical Commission (IEC) has developed Standard 1131, which defines the specifications required for languages that operate programmable controllers such as the PLC. [64]

The PLC can be designed as a standalone controller with hardwired mimic panel for the operator interface for local control of machineries in remote areas or as part of a DCS, TMS or SCADA systems. It also can be designed as a standalone PLC-based system with personal computer (PC) based HMIs for the operator interface console. Normally, a PLC-based PAS consists of multiple PLCs and operator HMI consoles and can be used for operating and controlling sequential process applications such as the cyclic sequence control of instrument air dehydrators and hydrocarbon desiccant catalysts regeneration in petrochemical and oil industries. Figure 5 shows a typical layout of a PLC-based PAS architecture and the hardwired connections from the field instruments up to the operation HMI consoles located at the CCR spanning through junction boxes, marshaling cabinets, I/O systems cabinets, Controller system cabinet, PIB, and CCR rack room. [68]

1.1.1 Junction Box

The field instruments include pressure, level, flow, and temperature switches, valve position limit switches, motor status for pumps/compressors/mixers, start/stop command pushbuttons, field panel indications, and start/stop and open/close output signals. Each field instrument is hard wired to the closest junction box in the range of 10 to 50 meters.

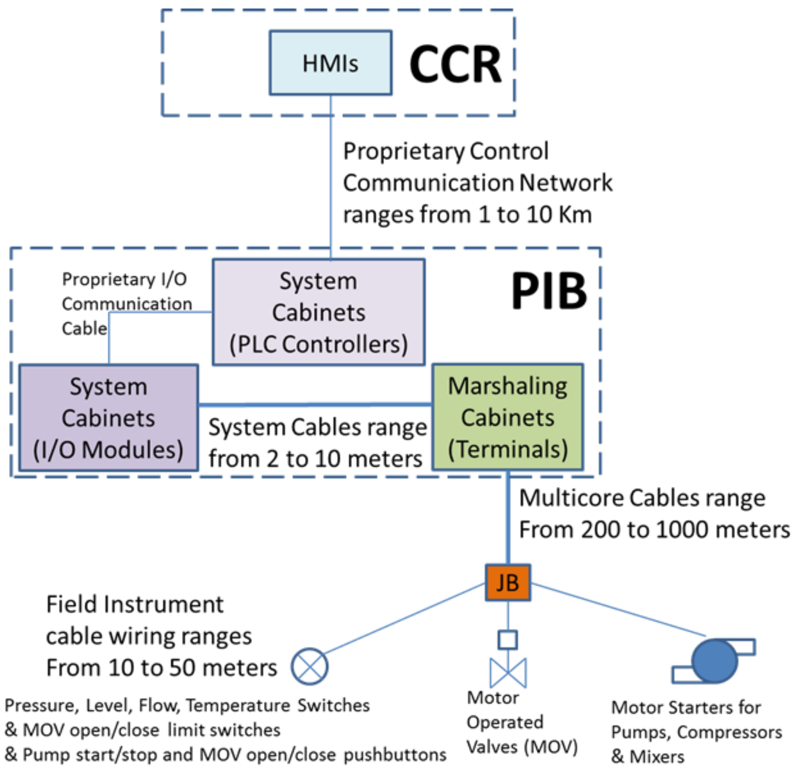

HMIs
CCR
Proprietary Control Communication Network ranges from 1 to 10 Km
Proprietary I/O Communication Cable
System Cabinets (PLC Controllers)
PIB
System Cabinets (I/O Modules)
System Cables range from 2 to 10 meters
Marshaling Cabinets (Terminals)
Multicore Cables range From 200 to 1000 meters
JB
Field Instrument cable wiring ranges From 10 to 50 meters
Pressure, Level, Flow, Temperature Switches & MOV open/close limit switches & Pump start/stop and MOV open/close pushbuttons
Motor Operated Valves (MOV)
Motor Starters for Pumps, Compressors & Mixers

Figure 5 - Typical PLC-Based PAS Layout

The junction box is an enclosure with terminal strips for connecting cables between field devices and PIB/CCR as shown in Figure 6. [55]

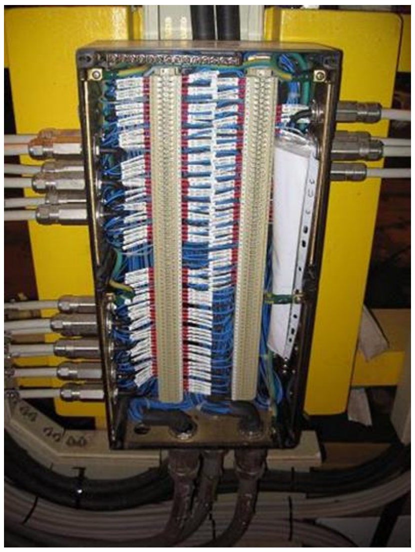


Figure 6 - Typical Junction Box

Each junction box is hardwired using multicore cables to the closest marshaling cabinet inside PIB/CCR in the range of 200 to 1000 meters.

1.1.2 Marshaling Cabinet

The function of the marshaling cabinet as shown in Figure 7 is to interface the incoming multicore cables with the I/O module connection and to perform the cross wiring function.

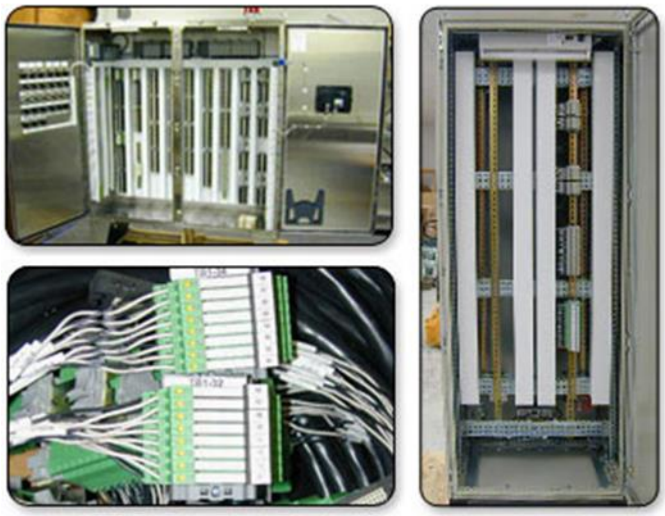


Figure 7 - Typical Marshaling Cabinet

Cross wiring is always necessary since the number of incoming field signals within multicore cables and the channel quantity of the I/O modules is always different. Another reason for the cross wiring is due to the mix of input and output field signals within the same incoming multicore cables and the requirement to split them into consecutive and dedicated terminals for the associated input and output modules terminals. The last reason for cross wiring is the requirement for routing the input and output signals to the designated PLC. Each marshaling cabinet is hardwired using prefabricated multicore system cables to the designated system cabinets inside PIB/CCR in the range of 2 to 10 meters. [60]

1.1.3 System Cabinet

The purpose of the system cabinet shown in Figure 8 is to provide terminals to interface with the marshaling cabinets and to house the PLC power supply, I/O modules, controller CPU, communication modules, engineering work station, and the auxiliary power supply for powering the field instruments. [34]

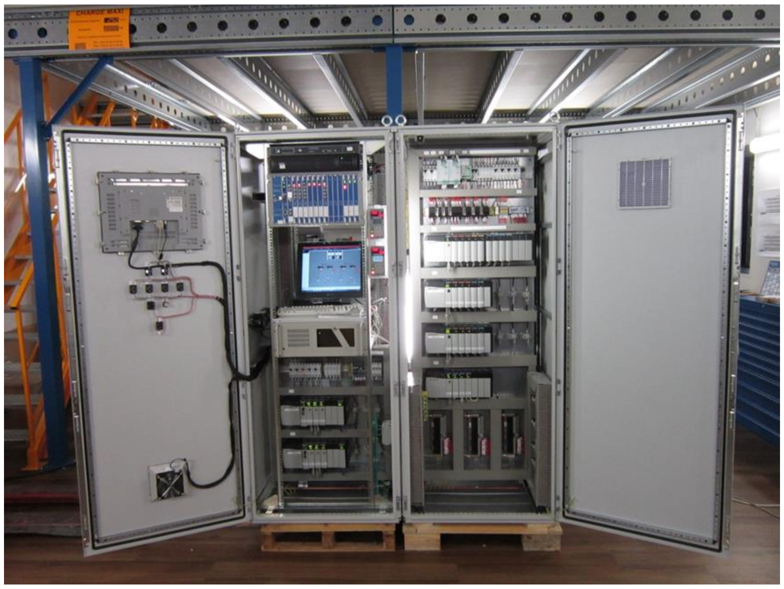


Figure 8 - Typical System Cabinet (Source: Rockwell Automation, Inc.)

1.1.4 Process Interface Building

The PIB is an explosive proof building used to house and protect the system and marshaling cabinets from deteriorating effects of the weather. It is unmanned and environmentally controlled building suitable to house delicate active electronics. Its location is carefully selected to withstand any unexpected field process explosion and to be as close as possible to large number of junction boxes in order to minimize the cost of cabling. [34]

1.1.5 Centralized Control Room

The CCR is a room serving as a central location for monitoring and controlling large manufacturing facilities or physically dispersed services. The CCR for vital facilities are typically tightly secured and manned continuously throughout the year. It has two major sections: rack room and control room. The function of the rack room is identical to the function of the PIB. The control room includes HMIs, engineering, and maintenance consoles as well as an auxiliary control panel for hardwired pull/push buttons for emergency and plant shutdown and annunciation panel for critical alerts. All consoles interconnected through the plant information communication network. The CCR plant information communication network is extended by network segments to connect all PIBs that range from 1Km to 10 Km in distance. [73]

1.2 PLC Evolution

During the 1960s, control systems were implemented using relay controller as shown in Figure 9. It lacked the flexibility for process changes or expansion and troubleshooting was difficult. [55] Any modifications required highly trained engineers for intensive hours of tedious work. Bill Stone part of the Hydramatic Division of GM presented a paper at the

Westinghouse Conference in 1968 outlining problems related to the reliability and documentation of the machines. It also presented design criteria to develop ”standard machine controller” which included: elimination of costly scrapping of assembly-line relays during model changeovers, reduction of machine downtime related to control problems, and a provision for modular future expansion.

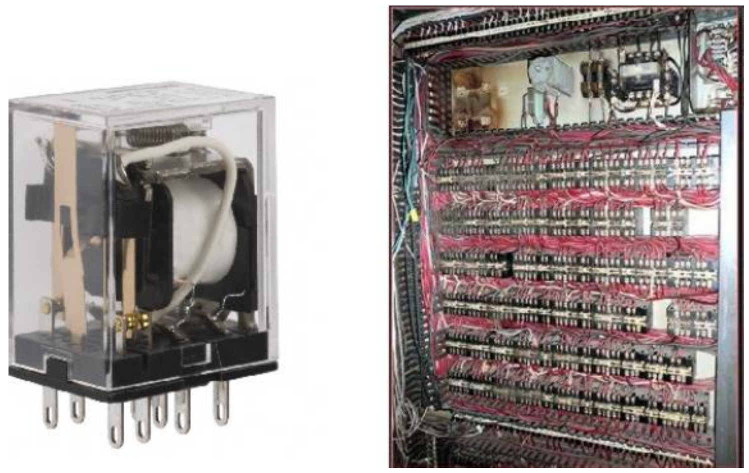
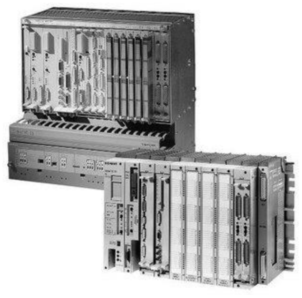
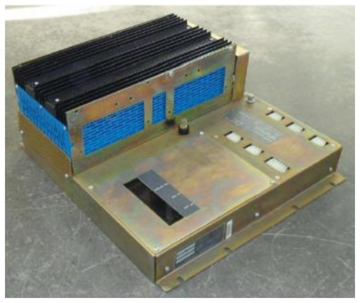


Figure 9 - Typical Relay Control Cabinet

A proposal request to build a prototype based on these specifications was given to four control builders: Allen-Bradley, Instruments, Inc., Digital Equipment Corporation, Century Detroit, and Bedford Associates. Digital Equipment brought ”mini-computer” and was rejected due to the problem of static memory. Allen-Bradley had two attempts: PDQ II and PMC. Both of them were too large, too difficult to program and too complex. Morley, Greenberg, Landau, Schwenk and Boissevain from Bedford Associates formed Modicon and built the Programmable Controller 084 in 1969, shown in Figure 10a. Greenberg and Rousseau from Modicon developed the Modicon 184 in 1973 as shown in Figure 10b, and Struger and Dummermuth from Allen-Bradley developed the Bulletin 1774 PLC (Patent#3,942,158) in 1974 as shown in Figure 10c., The acceptance of the PLC during its early days was very difficult; however, this technology made the constant rewiring of control panels obsolete. [74]





(a) (b) (c)

Figure 10 - Modicon 084 & 184 and Allen-Bradley 1774 PLCs

1.3 Research Goal, Scope and Contribution

Four key impediments dampens the aspired benefits of the overall PLC-based process automation solution; (1) the escalating capital cost for managing the life cycle of multi proprietary systems, (2) the lack of consistent and effective security protection at the control system level, (3) the low system utilization due to inherent conflicting constraints, and (4) the lack of real-time integration of heterogeneous process automation controllers.

The process automation systems run the gamut of age, technology, manufacturer, and family product series. As the equipment ages; components deteriorate, older technologies become superseded with newer products, manufacturers consolidate, product lines are eliminated, and technical support capabilities diminish. These reliability and obsolescence issues culminate in a broad range of challenges causing a huge escalating capital cost for managing the life cycle of these proprietary systems.

Each vendor implements its proprietary control system protection schemes ranging from no protection at all to physical key switches to protect the memory including the control application programs against any unauthorized personnel or any malicious intents. Managing many different and inconsistent security protection mechanisms is an overwhelming task and does not effectively reduce the risk of cyber security attacks.

Each PLC is constrained by one control application, maximum number of I/O modules, maximum size of CPU Memory, and CPU scan time resulting in overall low system utilization.

Most of the time, PLCs have proprietary control communication network and lack seamless real-time integration with distributed control system, supervisory control and data acquisition and/or the corporate enterprise resource planning system.

The objective of the PhD dissertation is to develop a new vendor independent and commercial off-the-shelf (COTS) virtually clustered automation platform (vCAP) to address the reliability and obsolescence challenges resulting from the life cycle management of proprietary PLC-based system while meeting the plant requirements and utilizing minimum system resources. The new vCAP is based on decoupling the controller from the I/O system capitalizing on emerging real-time Data Distribution Service (DDS) middleware technology as well as the evolution of high performance and fault-tolerant virtual computing and networking technologies.

The testing methodology for the new design has been accomplished as follows:

- A small scale prototype of the vCAP is implemented to test its performance compared to a conventional proprietary PLC.

- Software based emulation model is implemented in a large scale to demonstrate the performance sustainability of the vCAP while growing in size based on the number of field I/O signals and number of nodes.

The main potential tangible benefits of the new vCAP solution are:

- Overcome obsolescence challenges of proprietary PLC equipment.
- Reduce the initial capital cost for grass root automation systems by eliminating the need for PIBs, marshaling and system cabinets and associated cabling as well as the need for multiple proprietary standalone controllers and communication components.
- Minimize the life cycle capital cost to sustain the initial capital investment.
- Improve real-time system integration.
- Improve cyber security protection at the control level.

1.4 Dissertation Structure

A detailed literature review on PLCs and emerging middleware as well as virtual computing and networking technologies is described in chapter 2. Chapter 3 establishes the motivation and formulates the business and technical case of the new proposal. The design evolution and the architecture of the proposed vCAP solution are described in chapter 4. The research methodology and testing as well as the performance analysis of the proposed vCAP solution is detailed in chapter 5. Finally, chapter 6 summarizes the research conclusion and provides a future framework for detailing the vCAP design criteria.

CHAPTER 2

LITERATURE REVIEW

2.1 Process Automation Systems

Process is a sequence of interdependent and linked procedures. At every stage, one or more resources such as employee time and energy are consumed to convert inputs such as raw materials or parts into outputs or final products. The outputs then serve as inputs for the next stage until final products are reached. [15]

Process safety and control topics are crucial for designing and developing safe, reliable, effective, efficient, and sustainable processes.

Safety control systems are intended to prevent unsafe operation of the controlled equipment and can be classified into three categories: safety alarm systems, condition monitoring systems, and emergency shutdown systems. Safety alarm systems are used to detect the presence of fire smoke or leak of hazardous gases, and to alert people through audio and visual devices such as horns and beacons. The condition monitoring and equipment protection systems include vibration monitoring system, temperature monitoring systems, permissive and interlock systems, and machine protection systems. Emergency shutdown systems represent a layer of protection that mitigates and prevents a hazardous situation from occurring. [73]

Process control discipline deals with development of control strategies for maintaining the outputs of a specific process within desired ranges. It is extensively used in industry and enables mass production of consistent products. Automation or automatic process control is the application of control theory for regulating manufacturing processes without direct human intervention. The mathematical basis of the control theory started in the 18th century and advanced rapidly in the 20th century. The main advantages of automation are increased productivity, improved consistency and quality, and reduced operation cost. [27]

Process automation are classified into four categories based on the core technology utilized for developing the control strategy: mechanical, pneumatic, hydraulic, and electrical.

A mechanical system manages power to accomplish a task that involves forces and movement. Although mechanical systems possess many advantages including high durability and reliability, they are also subject to limitations. For example, With purely mechanical flight control systems, increases in the control surface area required by large aircraft or higher loads caused by high airspeeds in small aircraft lead to a large increase in the forces needed to move them by the pilots. Hence, the size of the aircraft and its performance are limited by the pilot's muscular strength. [15]

A pneumatic system uses compressed air to transmit and control energy. It consists of three parts: compressed air generator and storage tank, air cleaning and conditioning, controlling and operating section. Pneumatic systems have many advantages including high effectiveness, durability, adaptability to harsh environment, and safety. However, they are also subject to limitations. For example, the operation of the pneumatic systems is subject to the volume of available compressed air causing a decrease in the overall accuracy of the system. [21]

Hydraulic control systems use non-compressible fluid-based operation within a sealed system rather than electronics or pneumatic power. Many modern machines rely on either hydraulic controls or a hybrid electric-hydraulic system. One of the primary advantages to using hydraulic control systems is the ability to handle very large loads or accommodate tremendous forces. Hydraulic systems also allow for very precise and accurate handling in more specialized applications. However, the hydraulic fluid used within these systems can be highly corrosive, and may lead to extended maintenance and repairs over time. [34]

The electrical control systems include any controls that use electrical-based operation rather than pneumatic or hydraulic power. They can be further classified into three subcategories based on the core technology used in the control system design: relay control panel, solid-state logic control panel, and microprocessor-based control system.

Relay-based control systems are best suited for controlling simple and fixed processes that require less than 100 discrete input/output signals. The main disadvantages of relay-based control system are the lack of computation capabilities, the challenge in developing complex logic, the difficulty to expand and/or modify an existing control strategy, and the lack of communication capabilities to interface with processor based HMI consoles. [55]

Solid-state control systems are best suited for controlling simple and likely fixed processes in the foreseeable future that require less than 100 discrete input/output signals. The main disadvantages of solid-state control system are the lack of computation capabilities, the challenge in developing complex logic, the difficulty to modify an existing control strategy, and the lack of communication capabilities to interface with processor based HMI consoles. [73]

The microprocessor-based control systems can be further classified into five subcategories based on the core architecture of the control system design: PC-based control, PLC-based control, distributed control, supervisory control, and process automation.

PC-Based Control:

The architectural foundation layer of the PC-based controller is an industrially rugged PC hardware with a pre-emptive real time operating system. Built on this foundation is a suite of soft control modules including programmable logic control, continuous process control, supervisory control and data acquisition, and motion control. On top of those real time soft modules is an operator interface module which provides advanced HMI functions and network connectivity to Web, Local Area Network (LAN), major field bus and PLC controls. [69]

The following are the advantages of the PC-based controller:

- Consists of vendor independent hardware and software
- Utilizes manufacturer-independent standards for networking
- Cost effective solution

The following are the disadvantages of the PC-based controller:

- Applicable to only small and non-critical applications
- Limited scalability due to the requirement of having fixed centralized database
- Constrained I/O signals due to the limited number of expansion slots available in the PC motherboard

- Any intermittent hardware or software malfunctions in the PC will result in a total shutdown for the process operation
- Low reliability and availability due to the lack of continuous self-diagnostics testing and the lack of hardware redundancy
- Limited life span of the PC hardware
- Not suitable for harsh environment

PLC-Based Control:

A PLC is a specialized computing system used for control of industrial machines and processes. [74]

The following are the advantages of the PLC:

- Flexible where the original equipment manufacturers can modify the control application in the field and reapply to other systems quickly and easily
- Less and simpler hard wiring in comparison to the conventional relay control circuits
- Built based on solid-state components with no moving mechanical parts
- Based on modular plug-in construction allowing the options for mixing I/O modules and for repairing and expanding the system easily
- Based on sophisticated instruction sets that handle much more complicated applications than those can be implemented by relay-based and solid-state systems
- Easy to troubleshoot using resident self-diagnostic and override functions
- Less expensive compared to the relay control, except for small and fixed applications
- Communication capabilities with other controllers or computer equipment to perform supervisory control and data gathering

- Very reliable; PLC has no disk drive, CD drive, keyboard, or monitor and designed to operate in the industrial environment. PLC reliability can be improved using redundant components

The following are the disadvantages of the PLC:

- A proprietary system; the system must be upgraded or replaced prematurely if the original vendor discontinues manufacturing spare parts due to a change in its marketing strategies or the lack of third party subcomponents
- Requires skillful work force for maintaining the system; otherwise, troubleshooting and repair could take very long time to bring the system back online
- Not cost effective for simple and fixed applications
- Limited computing and complex instructions and capabilities

Distributed Control:
Distributed control systems are dedicated systems used to control manufacturing processes that are continuous or batch-oriented such as oil refining and petrochemicals. A typical DCS consists of functionally and/or geographically distributed digital controllers capable of executing from 1 to 256 or more regulatory control loops in one control box. The input/output devices can be integral with the controller or located remotely via a field network. Today’s controllers have extensive computational capabilities. DCSs are usually designed with redundant processors to enhance the reliability of the control system. Most systems come with displays and configuration software that enable the end-user to configure the control system without the need for performing low-level programming, allowing the user also to better focus on the application rather than the equipment. However, considerable system knowledge and skill is required to properly deploy the

hardware, software, and applications. Many plants have dedicated personnel who focus on these tasks, augmented by vendor support that may include maintenance support contracts. [33]

The following are the advantages of the DCS:

- Improved scalability; unlike a centralized system and common database, in which the amount of data that can be stored depends on the limitations of one host and one database
- Improved performance; local user databases distribute the load on system resources, reduce network traffic, and eliminate communication bottlenecks
- Increased availability by adding redundant components
- Flexibility in system design and incremental growth; modularity and ease of expansion where controlling power can be added in small increments
- Reduced risk by distributing the control function among number of small controllers
- Ease of operation and maintenance
- Overall optimization; ease of monitoring of more plant parameters while ensuring tighter control on them
- Economic of scale; distributed controllers offer a better price/performance than centralized systems
- Speed; distributed systems may have more total computing power than centralized systems

The following are the disadvantages of the DCS:

- Proprietary system that must be upgraded or replaced prematurely if the original vendor discontinues manufacturing spare parts due to a change in its marketing strategies or the lack of third party subcomponents
- Requires skillful work force for maintaining the system; otherwise, troubleshooting and repair could take very long time to bring the overall system back online
- Inherent security issue
- Hindering data transmission during up normal network malfunctions causing communication networks saturation
- Not cost effective for simple or pure sequential process applications

Supervisory Control:

Supervisory control and data acquisition systems are centralized systems which monitor and control entire sites, or complexes of systems geographically spread out over large areas, anything from an industrial plant to a nation. The major function of SCADA is for acquiring data from remote devices and providing overall control remotely from a SCADA host software platform. Host control functions are usually restricted to basic overriding or supervisory level intervention. [73]

SCADA systems are significantly important systems used in national infrastructures such as electric grids, water supplies and pipelines. However, they may have security vulnerabilities, so the systems should be evaluated to identify risks and solutions implemented to mitigate those risks. [26]

The following are the advantages of the SCADA system:

- Reduce the operating and maintenance costs; through the deployment of a centralized SCADA system, one can significantly reduce operating and maintenance costs; fewer personnel are required to monitor field equipment in remote locations, resulting in increased operator effectiveness; and less maintenance trips are required, resulting in decreased maintenance and training costs
- Flexibility and scalability; the user can start out at a low level of capital investment and can incrementally grow in size to connect thousands of sensors over a wide area and can record and store a very large amount of data

The following are the disadvantages of the SCADA system:

- SCADA system are based on PLCs and RTUs that are proprietary and must be upgraded or replaced prematurely if the original vendor discontinues manufacturing spare parts due to a change in its marketing strategies or the lack of third party subcomponents
- Limited extensibility to new applications such as real-time safety alarm systems
- Inherent security issue

Process Automation:

A process automation system, similar to DCS, is used to automatically control processes of oil and gas and petrochemical industries. However, the PAS is mostly based on open standard networks to interconnect sensors, controllers, operator terminals and actuators in contrast to a DCS which is traditionally proprietary. The main building block of PAS, DCS, and SCADA systems is the microprocessor-based programmable controller evolved from the PLC technology. The microprocessor-based programmable controller ranges from the simplest controller called RTU and used mostly in data acquisition systems to the sophisticated DCS controller used in distributed control systems. In between, the basic PLC is used mostly in manufacturing automation and supervisory control systems. In other words, the PLC concept has evolved with much less control functionalities to form the cost effective RTU system and it has evolved with continuous and advanced regulatory control functionalities and complete hardware redundancy to form the highly available DCS controller.

2.2 PLC Architecture Evolution

The first PLCs were small, with a target to replace stand-alone automotive manufacturing relay control panels. The PLC features and capabilities increased gradually to cover all sequential control applications. The PLC architecture has evolved to address the design requirements of each application regarding the system availability, scalability, reliability flexibility, maintainability, and testability. The first PLC architecture was the fixed PLC style. [74]

2.2.1 Fixed PLC

This PLC type is generally used for basic functions and is not suited for future expansion. The fixed PLCs are manufactured in different sizes based on the type and number of I/O channels. The fixed PLC is not flexible when the I/O electrical signals of replaced field instruments change to a different voltage and the entire unit may need to be replaced. To overcome the fixed PLC system maintainability and flexibility limitations, a modular PLC was developed. [13]

2.2.2 Modular PLC

Modular PLC allows for installation of fixed number of I/O modules, or printed circuit boards contained inside a casing. There are different sizes of chassis based on the number of I/O module slots ranging from three to sixteen slots. Each slot can be allocated for either input or output module. Each module is designed for a specific voltage and can handle 4, 8, 16, or 32 channels. Modular PLCs offer simpler troubleshooting, ultimately alleviating system downtime. In other words, each I/O module can be replaced online, with no need to power off the entire system, and can be sent to the vendor for troubleshooting and repair while the PLC system continues its normal function.

The PLC system provided flexibility in programming to modify the control strategies easily compared to the conventional hardwired relay control panel. This flexibility in PLC system allowed for more sophisticated automation applications and often required additional input/output electrical signals. This enhancement would not be captured if there are no available I/O channels to accommodate the new requirement. This limitation in scalability is facing both fixed and modular PLC systems where there is enough capacity in the PLC CPU for additional automation enhancement, but constrained by the fixed PLC I/O size. The impact of the PLC system scalability limitation is worse in the fixed PLC version. The fixed PLC may need to be replaced completely. However, the modular PLC may need to replace the chassis only with a larger one while maintaining the existing CPU and I/O modules. To overcome the PLC system scalability limitation, PLC I/O expansion was developed for both the fixed and modular PLCs. [74]

2.2.3 PLC with Local I/O Expansion

The fixed PLC can be expanded by extending the communication bus from the base unit to the fixed I/O expansion units in daisy chain. The modular PLC can be expanded by extending the main backplane communication bus to an I/O expansion chassis. The scalability limitation was improved using the bus expansion strategy for both the fixed and modular PLCs. One additional scalability improvement for the modular PLC was achieved using remote I/O expansion units in addition to the main backplane communication bus expansion. [15]

2.2.4 PLC With Remote I/O Expansion

The rise in complexity of machines and systems was placing increasing demands on automation technology. Branched sub-systems needed to be precisely integrated into complex topologies. A rapid and reliable transmission of signals and data was critical for smooth operation. This need mandated the evolution of the modular PLC to include remote I/O expansion using proprietary remote I/O communication network for integrating the electrical signals far away from the main PLC rack. The evolution of providing remote I/O expansion equivalent to the extension of the main local rack created an opportunity to improve the reliability of the modular PLC system used for controlling mission-critical processes. [71]

2.2.5 Redundant PLC Architecture

The evolution is to use two modular PLCs, connected in a Hot-Standby redundant configuration, with only remote I/O expansion units. The use of this architecture is required for applications in harsh environments and critical facilities where even short outages are not tolerated. With redundancy on all system levels of the primary and the secondary modular PLCs including the CPUs, chassis, power supplies, system software, and the application program, this architecture provides maximum reliability. The Hot-Standby redundant configuration allows both modular PLCs to receive the input status from the remote expansion units and only the primary modular PLC can communicate to the remote I/O expansion units for controlling the output signals. Any failures in the primary modular PLC will result in a smooth switchover to the secondary modular PLC that becomes the primary one for controlling the output signals, i.e. no deviation between output values of both the failed and the new primary modular PLCs. This redundant PLC architecture has addressed the design requirements of each application regarding the system reliability, scalability, availability, flexibility, maintainability, and testability.

The system failure, abbreviated as “F”, refers to the probability that the system will be down. Assuming that the reliability of one modular PLC is “R”, the probability that the simplex modular PLC system will be down is F = 1 – R. However, for a redundant modular PLC system, the probability that the system will be down is $F = (1 – R)^2$. Figure 11 shows the reliability of the redundant modular PLC system in comparison to the reliability of the simplex modular PLC system.

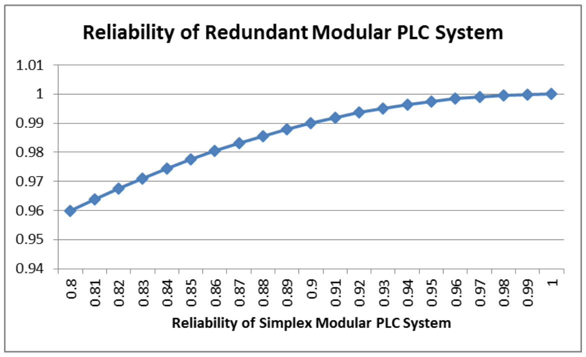


Figure 11 - Reliability of Redundant Modular PLC System

Obviously, higher reliability is achieved using the redundant modular PLC system. When R = 0.9, 0.99, 0.999, the redundancy will improve the reliability to 0.99, 0.9999, 0.999999 respectively. In other words, the number of 9s will be doubled.

The scalability of the redundant modular PLC system is achieved using remote I/O expansion where remote I/O chassis can be added to a maximum of 31 chassis' in a daisy chain manner.

The definition of system availability abbreviated as "A" is equal to the up time divided by total time. The up time is equal to the total time minus the down time. For example, assume that the repair time for a failed modular PLC is "RT", the total process time duration is "TT", and only one PLC system failure jeopardized the process operation happened during the total processing time. Hence, the down time is equal to the repair time and the availability of the simplex modular PLC system is = (TT – RT) / TT. For the redundant modular PLC system, the availability is = 1 since there was only one failure and it was masked by the redundant configuration. Therefore, the system availability has improved using the redundant modular PLC system and it is proportional to RT as shown in Figure 12, assuming that the total time of continuous process operation is one year.

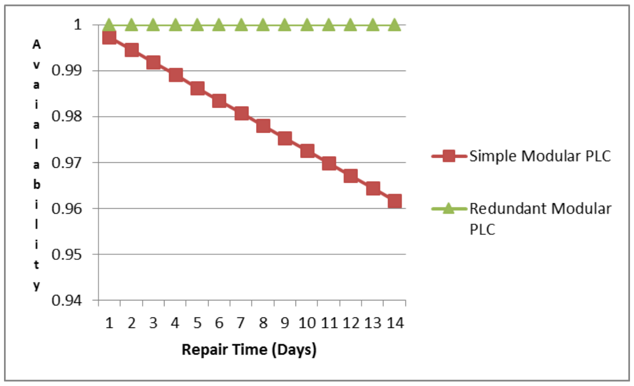


Figure 12 - Redundant Modular PLC System Availability

The flexibility of the redundant modular PLC system is achieved using modular I/O system with different density I/O modules (4, 8, 16, 32 channels) and extensive types of discrete I/O modules including AC and DC cards ranging in voltage potential from 5 to 220 volt and analog I/O modules including 0-20 mA, 4-20 mA, 0-10 volt, and 0-5 volt. The I/O system contains pluggable sections so that modules can be mixed and matched.

The redundant modular PLC system maintainability and testability are superior due to the availability of redundancy in the main PLC system, the modularity of the I/O system, and the capability of online replacement of any system modules without impacting the remaining of the PLC system. Troubleshooting and testing of faulty modules can be done offline without the need of the running PLC system. [55]

2.3 PLC HMI Evolution

First installations of small PLCs targeted to replace stand-alone automotive manufacturing relay control panels such as the interlock systems for process equipment protection did not require any operation intervention. However, the PLC applications have evolved to include sequential control strategies that require operation intervention using operator interface devices. The following describes the evolution of the operator interface devices.

2.3.1 PLC Mimic Panel

The PLC mimic panels are primarily used in power generation, plants assembly line, chemical plants, power stations, cement plants, and chemical plants. As shown in Figure 13, the mimic panel includes schematic representations of complex systems for the operator to monitor and control the process. It also includes open/close, start/stop, on/off, pushbuttons, selector switches, indication lights, alarm annunciation panel, chart recorders, and analog meters. The electrical signals from the mimic panel are hardwired to the PLC discrete and analog I/O modules. The primary cost driver of the mimic panel is associated with I/O modules and wiring. [60]

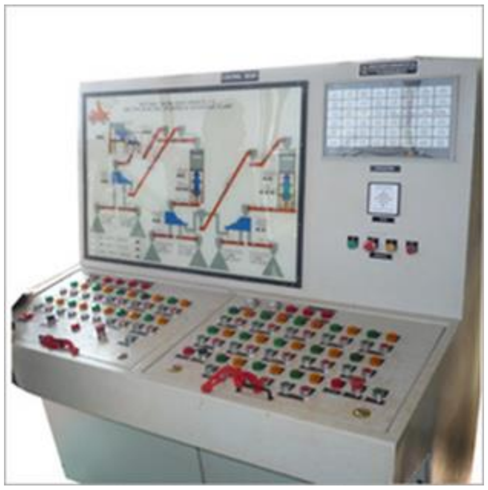


Figure 13 - PLC Mimic Panel

2.3.2 PLC Panel View

In order to minimize the cost of the PLC interface panels, processor based panel view displays were developed. The panel view display is designed to communicate to the PLC as a remote I/O chassis using built-in remote I/O communication adapter. It can be configured in software to include required schematic representations of complex systems similar to the mimic panel for the operator to monitor and control the process. It also includes software configured open/close, start/stop, on/off, pushbuttons, selector switches, indication lights, alarm annunciation panel, chart recorders, and analog meters using either display with membrane keypad or directly through a touchscreen display as shown in Figure 14. This type of panel emulates a remote chassis of I/Os and eliminates the use of PLC I/O modules and wiring for the operator interface devices; however, the panel view remote I/O communication adapter consumes a complete logical rack out of the 31 space available. In other words, all software configured I/O points in the panel view are taking equivalent space from the PLC I/O map similar to those utilized by the hardwired I/O modules in the mimic panel. Therefore, the required PLC size is determined based on the number of actual hardwired I/O signals as well as the number of software based panel view I/O signals. [68]

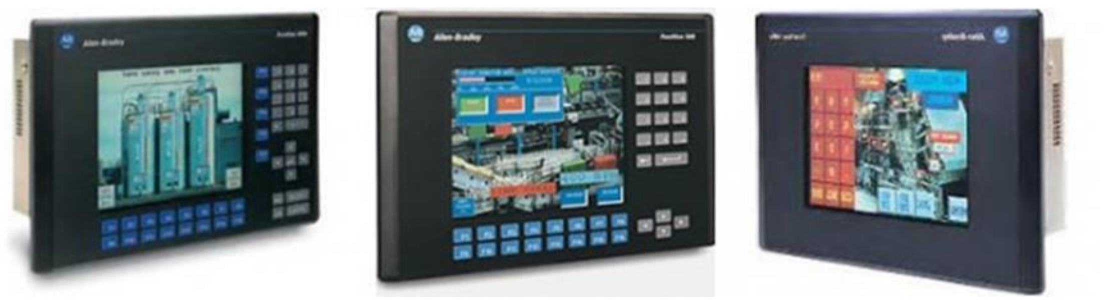


Figure 14 - PLC Panel View (Source: Rockwell Automation, Inc.)

2.3.3 PLC HMI Workstation

In order to decouple the mimic panel remote software I/O points from the actual PLC I/O map, an industrial PC based HMI workstation is developed to communicate directly to the PLC through defacto standard communication link such as Modbus or Ethernet. The HMI graphic schematics, pushbuttons for open/close, start/stop, and on/off commands, selector switches, indication lights, chart recorders, and analog meters can be configured to allow the operators to easily monitor and control the process through a direct communication interface to the PLC system with a minimum requirement of one-second data refresh rate. With a single HMI workstation, a dedicated hardwired annunciation alarm panel is still required to ensure visual monitoring of all critical alarms continuously while navigating through the HMI graphic schematics. The HMI workstation can be offered using either display with membrane keyboard or directly through a touchscreen display as shown in Figure 15. [72]

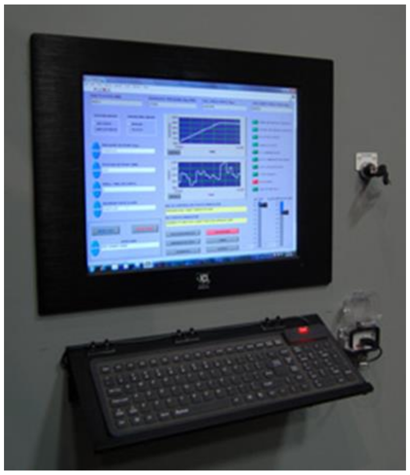


Figure 15 - PLC HMI Workstation

2.3.4 PLC HMI Console

With the advancement in PC development and in order to eliminate the need of additional dedicated hardwired annunciation alarm panel along with the HMI workstation, HMI console is developed. The HMI console idea is similar to the HMI workstation concept with multiple displays connected to a single PC as shown in Figure 16. Along with the main HMI display for navigating through the graphic schematics, a dedicated display is configured for displaying critical alarms and a another dedicated display for displaying alarm summary and sequence of events.

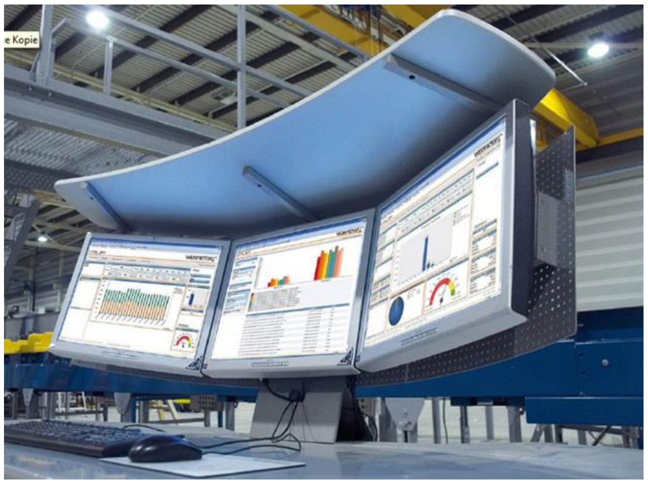


Figure 16 - PLC HMI Console

With HMI console concept, the operator interface device became completely independent from the PLC system design. As a standalone system, interfacing the PLC system to HMI consoles is very cost effective. However, where there is an existing DCS or SCAD system in the processing facility, the PLC HMI console becomes redundant and costly. Hence, integrating the PLC system to the existing DCS or SCADA system becomes cost effective and more efficient for the operator to monitor and control the overall process facility through a single and standard view window. [24]

2.4 Virtual Fault-Tolerant Servers

The traditional approach of installing one physical server per application results in low server utilization rate, the fraction of total computing resources engaged in useful work. Server virtualization offers a way to consolidate controllers by allowing multiple different control applications run on one physical host server as shown in Figure 17.

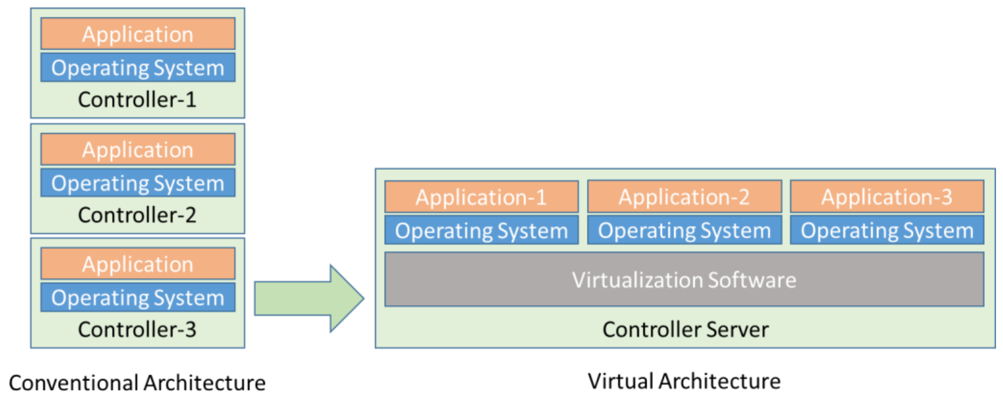


Figure 17 - Server Virtualization

Therefore, instead of operating many controllers at low utilization, virtualization combines the processing power onto fewer servers that operate at higher total utilization. In addition, virtualization improves scalability, reduces downtime, and enables faster deployments.

However, it magnifies any physical failures. Hence, for critical mission applications utilizing server virtualization, it is crucial to increase the system reliability and availability of the shared physical servers because any faults could jeopardize the operation of all virtual servers. [7]

Single fault tolerance property enables the system to continue its intended operation properly during the event of having a single fault in any subcomponents. The fault tolerance attribute is essential for improving the system availability of virtualized mission critical applications. There are two methods in developing single fault tolerance servers either based on software such as vSphere developed by VMware or based on hardware such as GeminiEngine developed by NEC. [28,47]

2.5 Real-Time Middleware Integration Model

The correctness of computations in a real-time system depends on their logical correctness and the time at which the result is produced. Real-time middleware integration model illustrated in Figure 18, is the communication model required for interconnecting disparate subsystems in order to timely leverage applications collaboration across the overall system for improving business efficiency, enhancing scalability, and maximizing the return on capital investment. It facilitates interoperability among application programs, networks, and servers, thus masking differences or incompatibilities in hardware architecture, operating systems, databases, and programming languages. [4]

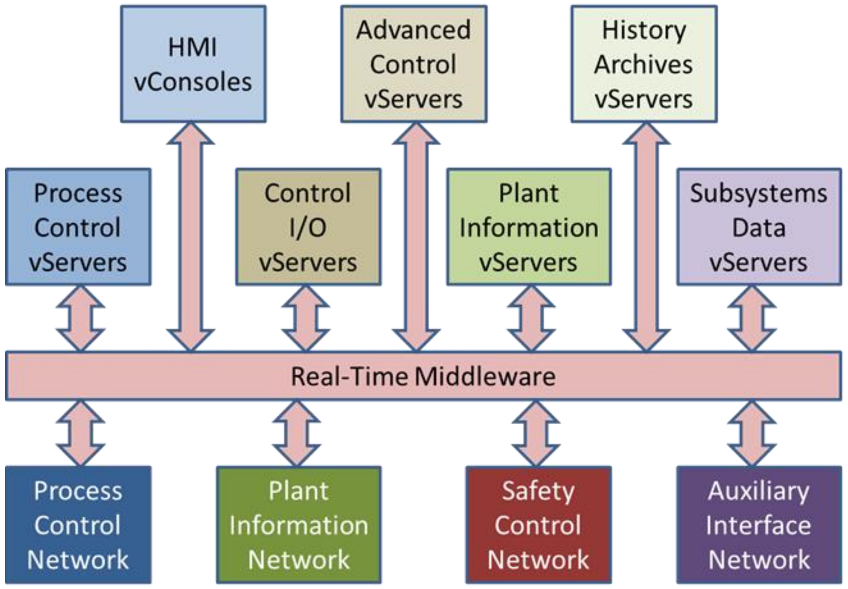


Figure 18 - Real-Time Middleware Integration Architecture

2.5.1 Real-Time Systems

Real-time systems can be classified into six categories based on the consequence of missing a deadline: hard, weakly hard, firm, soft, near, and non-real-time systems.

Hard real-time systems do not tolerate missing a deadline and when it happens, it is considered a total system failure and could lead to potential loss of life and/or big financial damage. Process automation systems for controlling hydrocarbon processing facilities are good examples of hard real-time requirement. [32]

Weakly hard real-time systems tolerate predictable *m* out of *n* deadline misses that may result in a minimum level of quality of service. Feedback control application is a good example of weakly hard real-time requirement where the control becomes unstable with too many missed control cycles. [31]

Firm real-time systems tolerate infrequent deadline misses, but the result is obsolete and may degrade the systems quality of service. In other words, the usefulness of a result is zero after its deadline. Weather forecasting system is a good example of firm real-time requirement. [1]

Soft real-time systems tolerate deadline misses, but these are not desirable and may degrade the systems quality of service. The usefulness of a result degrades after its deadline. Live audio-video system is a good example of soft-real-time requirement where violation of constraints results in degraded quality, but the system can continue to operate. [49]

Near real-time systems tolerate time delay in deadlines introduced by automated data processing or network transmission. It also refers to delayed real-time transmission of voice and video. It allows playing video images, in approximately real-time, without having to wait for an entire large video file to download. [56]

Non-real-time systems accomplish tasks with best effort quality of service and without specific deadlines. This means that a non-real-time system has no timing constraints that must be met to avoid failure. E-mail system is a good example of non-real-time requirement. [20]

2.5.2 Middleware Technology

Middleware is a collection of technologies and services to enable the integration of subsystems and applications across an overall system. The usage of the middleware technology can be classified into three categories: data integration, vendor independence, and common facade. Data integration improves interoperability by ensuring that exchange of information from heterogeneous data sources in multiple subsystems is kept consistent and appears to a user or system as a single, homogeneous data source. Vendor independence improves portability by abstracting the complexity and heterogeneity of the underlying distributed environment with the depth of network technologies, hardware, operating systems and programming languages so that even if one of the subsystems is replaced with a different vendor's equipment, the associated applications do not have to be re-implemented. Common facade provides a single consistent access and graphical interface to all subsystems and applications and shields operators and users from having to learn to use many different interface applications. The following describes the main characteristics of the middleware technology.

2.5.2.1 Middleware Characteristics

The purpose of using middleware is to isolate the application from the platform specific differences and provide facilities to hide the undesirable aspects of distribution transparency mechanisms. These mechanisms can be classified into the following different aspects [40]:

- Location Transparency: masking the physical locations from services
- Access Transparency: masking differences in representation and operation of the invocation mechanisms
- Concurrency Transparency: masking overlapped execution
- Replication Transparency: masking redundancy of the resources
- Failure Transparency: masking recovery of services after failure
- Resource Transparency: masking changes in the representation of a service and resources used to support it
- Migration Transparency: masking movement of service from one application to another
- Federation Transparency: masking administrative and technology boundaries

Properly developed and deployed middleware can reduce the task of developing distributed applications and systems by helping to provide a set of capabilities closer to the application design level abstractions. Hence, it can ease the integration and interoperability of software over diverse heterogeneous and separated environments by providing industry-wide standards for higher-level abstraction of portable software. Furthermore, it manages system resources by using higher-levels of abstractions and avoiding low level, tedious and error-prone platform details. It also reduces system lifecycle costs by building trusted reusable software patterns and providing a wide array of ready to use services for developers. [67]

Several standardization efforts are ongoing in several areas of middleware technology. These efforts have resulted in different classifications of the middleware solutions.

2.5.2.2 Middleware Classification

2.5.2.2.1 Heterogeneity-Based Classification

A broader approach for classifying middleware can be defined according to the type of heterogeneity related to hardware platforms, programming languages, and network connectivity. [61]

2.5.2.2.2 Non-Functional Requirements-Based Classification

A middleware taxonomy with respect to non-functional requirements depends on the fact that middleware is a distributed system and needs to meet the requirements of any type of distributed system. So any middleware can be evaluated and classified according to whether, and in what degree, it can meet each individual requirement. These requirements are hardware and software heterogeneity, openness, scalability, failure handling, security, performance, adaptability, and feasibility related to resource constraints. [23]

2.5.2.2.3 Layered-Based Classification

The Object Oriented Middleware decomposes middleware into multiple layers, shown in Figure 19. The host infrastructure layer encapsulates native operating system communication and concurrency mechanisms to create portable and reusable network programming components. These components eliminate error-prone and non-portable aspects of developing networked applications utilizing low level operating system programming. The distribution layer enables the invocation of operations from target objects regardless of location, operating system platform and communication protocols. The common services layer focuses on allocating, scheduling, and coordinating various end-to-end resources throughout the distributed system using a component programming and scripting model. The domain specific services layer is tailored to the requirements of a particular distributed real-time embedded systems domain. [42]

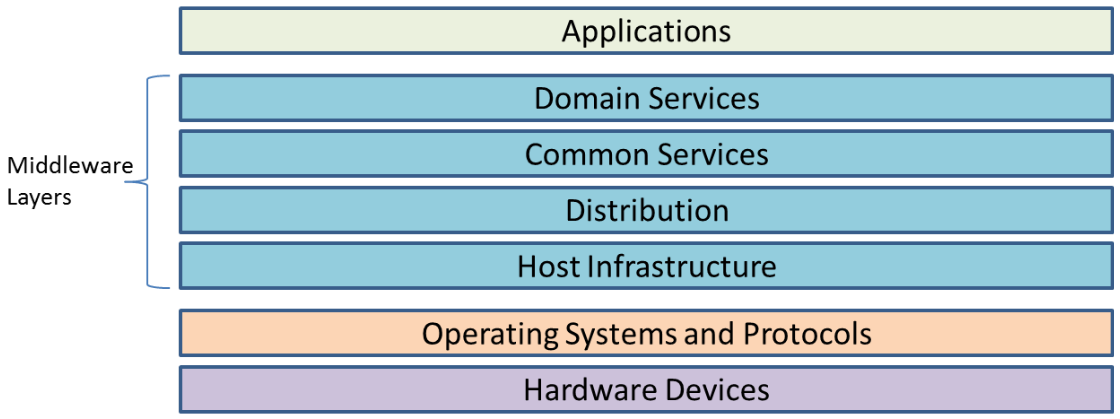


Figure 19 - Middleware Layers

2.5.2.2.4 Architectural-Based Classification

A classical classification of middleware solutions categorizes them according to their design and architectural elements used to build them; remote procedure calls, transaction-oriented, message-oriented, object-oriented, and component-oriented middleware. [39]

Many architectural paradigms and models are used in the development of middleware systems and can be classified according to their level of abstraction.

2.5.2.3 Middleware Architectures

The following describes the most widely used middleware architectures.

2.5.2.3.1 Message Passing Middleware

Message passing is the basic approach for inter-process communication, where data messages are exchanged between two processes, a sender and a receiver. A direct application of this approach is the socket application-programming interface. [19] As the message passing paradigm is at the bottom level of distributed systems paradigms, designers of real-time middleware based on this paradigm are usually overly involved with their immediate target platform to make reasonable use of its communication features in order to get the most valuable quality of service for this platform. This makes this middleware non-portable to other platforms. To achieve the portability in real-time applications and communication middleware implemented based on the message passing paradigm; researchers at Mississippi State University published real-time message passing interface standard, version 1.1 in 2002. [70]

2.5.2.3.2 Client-Server Middleware

The client-server middleware is the most common paradigm for distributed applications, and many other paradigms are built upon it. In this paradigm, asymmetric roles are assigned to two collaborating processes; server and clients. The server acts as a service provider, which waits passively for the arrival of requests from clients. The clients issue specific requests to the server and await its response. This middleware is the principal paradigm for the Internet-based client-server applications. [75]

2.5.2.3.3 Peer-To-Peer Middleware

In the peer-to-peer paradigm, the participating processes play equal roles, with equivalent capabilities and responsibilities. In this paradigm, each participant may issue a request to another participant and receive a response. This is different from the client server paradigm where there is no provision for server process to initiate communication. [44]

2.5.2.3.4 Message System Middleware

This paradigm is an elaboration of the basic message-passing paradigm. In this paradigm, a message system serves as an intermediary among separate, independent processes. The message system acts as a switch through which processes exchange messages asynchronously in a decoupled manner. A sender deposits a message with the message system, which forwards it to a message queue associated with each receiver. Once the message is sent, the sender is free to move on to other tasks. [81,140]

This paradigm can be classified into point-to-point and publish/subscribe messaging models. The point-to-point message model handles messages intended for a single receiver. Within a point-to-point message system, the messaging provider establishes queues to help ensure that a message is delivered to only one receiver only once. [58] Publish-and-subscribe systems handle messages intended for multiple receivers. Applications interested in the occurrence of a specific event may subscribe to messages for that event. When that waited event occurs, the process publishes a message, announcing the event or topic. The message system is responsible for distributing the message to all the subscribers. This model offers a powerful abstraction for multicasting or group communication. [25]

One of the most important efforts to build a real-time publish-and-subscribe based middleware is the Data Distribution Services middleware. DDS is a formal standard middleware specification published by the OMG group. [62] DDS targets mainly real-time systems; the API and quality of service (QoS) polices are chosen to balance predictable behavior and implementation efficiency and performance. [36] DDS is built on the idea of global data space of data objects that any entity can access, where the data object is uniquely identified by its keys and topics and the global data space itself is identified by its domain id; each publisher/subscriber must belong to the same domain to communicate. [46]

Each one of these entities can be configured through a corresponding specialized set of QoS policies, notified of events, and support conditions that can be waited upon the application. Since the DDS discovery is spontaneous and decentralized, the topics can dynamically change over the lifetime without any administrative impact, where end-points are discovered automatically, and dynamic dataflow established in a plug-and-play fashion, which means DDS is suitable for scalability. As DDS is targeted for real-time systems, its design supports a set of features, in a form of QoS policies, to enhance its performance. [35]

2.5.2.3.5 Remote Procedure Call Middleware

Remote procedure call (RPC) is an inter-process communication that allows a computer program to cause a subroutine or procedure to execute in another address space commonly on another computer on a shared network without the programmer explicitly coding the details for this remote interaction. This middleware was generated as a solution for the requirement of an abstraction that allows distributed software to be programmed in a manner similar to conventional applications running on a single processor. [50]

2.5.2.3.6 Distributed Object Middleware

In centralized object oriented software development, the software system can be built as a set of objects, where these objects communicate and coordinate among themselves on the same machine, in order to provide the required functionality of the system. The distributed object middleware was introduced as a natural extension of object oriented software development, by enabling the communication among objects that reside on different machines in order to build a distributed system. The distributed object middleware is built upon the client-server paradigm and can be represented by several models; remote method invocation, network service, and object request broker middleware. [48]

The remote method invocation middleware is the object-oriented equivalent of the remote method call. A process invokes the method in an object, which may reside in a remote host. [53]

In the network service middleware, service providers register themselves with directory servers on a network. A process desiring a particular service undertakes the following steps; (a) it contacts the directory server at run time; then (b) if the service is available on it, it will be provided a reference to the service; finally in (c) the requestor can access the service using this reference. In this manner, this paradigm is an extension to the remote method call paradigm. The difference is that the service objects are registered with a global directory service, allowing them to be looked up and accessed by service requestors on federated network. [11]

The object request broker middleware allows program calls to be made from one computer to another via a computer network, providing location transparency through remote procedure calls. It promotes interoperability of distributed object systems, enabling such systems to be built by piecing together objects from different vendors. [6] This paradigm is close to the remote method invocation paradigm. The difference is that this model acts as an intermediary which allows the object requestor to potentially access multiple remote or local objects, even when those objects were heterogeneous. [38]

2.5.2.3.7 Component Oriented Architecture Middleware

Component-oriented architecture is based on a set of well-known object-oriented programming practices such as encapsulation, and separation of concerns. Applications built using this paradigm, are decomposed into components that have clearly defined responsibilities and interact with one another through standardized interfaces. [30]

2.5.2.3.8 Application Server Middleware

The application server model is known as the second-generation client-server architectures or three-tier architecture model. The application server is a middle tier layer responsible for providing the access to objects or components to the clients requesting it. Web Services adopts this middleware to provide the next wave of web based computing. [37]

2.5.2.3.9 Tuple Space Middleware

In this middleware, a provider places objects as entries into an object space, and requesters, who subscribe to the space, access these entries. The object space provides a virtual space or meeting room among providers and requestors of network resources or objects in a manner that hides the details involved in resource or object lookup needed in other paradigms. Several models can be used to implement this middleware in distributed memory systems including centralized processing node, hashing allocation, partitioned structure, and fully distributed system. The Tuple space paradigm was first proposed as a part of the Linda coordination language for parallel and distributed processing, where the data is represented by elementary data structures called tuples, in a form of shared object, and the shared memory is a multi-set of tuples called tuple space. [66]

2.5.2.3.10 Collaborative Application Middleware

In systems such as virtual organization and video conferencing, multiple parties collaborate in order to provide a particular service. These systems use the collaboration paradigm to implement the required sharing of a common state. According to the placing of the common state of the system, the collaboration can be implemented by using messages to propagate the state to each local copy of the shared state, or by keeping the shared state in a central location where collaboration parties can access it. [29]

2.5.2.3.11 Enterprise Messaging System Middleware

An enterprise messaging system is a set of published enterprise-wide standards that allows organizations to send semantically precise messages between computer systems. This middleware promotes loosely coupled architectures that allow changes in the formats of messages to have minimum impact on message subscribers. [14]

2.5.2.3.12 Message Broker Middleware

The purpose of a broker is to take incoming messages from applications and perform some action on them including routing messages, transforming messages, aggregating messages, and augmenting messages. [41]

2.5.2.3.13 Enterprise Service Bus Middleware

An enterprise service bus is a software architecture model used for designing and implementing communication between mutually interacting software applications in a service-oriented architecture. Its primary use is in enterprise application integration of heterogeneous and complex landscapes. The concept has been developed in analogy to the bus concept found in computer hardware architecture combined with the modular and concurrent design of high-performance computer operating systems. [43]

2.5.2.3.14 Intelligent Middleware

Intelligent middleware provides real-time intelligence and event management through intelligent agents. It manages the real-time processing of high volume sensor signals and turns these signals into intelligent and actionable business information. The actionable information is then delivered in end-user power dashboards to individual users or is pushed to systems within or outside the enterprise. [28,98]

2.5.2.3.15 Content-Centric Middleware

Content-centric middleware offers a simple *provider-consumer* abstraction through which applications can issue requests for uniquely identified content, without worrying about where or how it is obtained. [2]

2.5.2.3.16 SQL-Oriented Data Access Middleware

SQL-oriented data access is middleware between applications and database servers. [76]

2.5.2.3.17 Embedded Middleware

Embedded middleware provides communication services and software/firmware integration interface that operates between embedded applications, the embedded operating system, and external applications. [3]

2.5.2.3.18 Transaction Processing Middleware

Transaction processing middleware provides a complete environment for transaction application that access relational database. In this middleware, clients call remote procedures stored on the server, which contain a set of SQL statements, transaction. [5]

2.6 Partial Solutions To Address Obsolescence Challenges

In the literature, there are several attempts to address process automation obsolescence challenges spanning most process automation functional levels.

The plant supervisory functional level has the shortest component life span related to the proprietary human machine interface consoles. The evolution of reliable HMI consoles based on standard UNIX and Windows operating systems for standalone PLC-based applications along with the strong desire from automation users to standardize the hardware of the DCS HMI consoles influenced DCS vendors to offer standard PC-based HMI workstations as an option along with their proprietary HMI consoles at the supervisory functional level. As a result, the cost of the HMI console was reduced by at least 50% and the HMI hardware obsolescence challenges were resolved based on utilizing standard PC workstations from multiple suppliers. [45]

The direct control functional level includes the controller and associated I/O modules. The I/O part accounts for about 75% of the total control system. The evolution of Foundation Fieldbus attempted to address the obsolescence challenges at this functional level by introducing two standards; H1 bus and High Speed Ethernet network. The objective of the H1 bus standard is to replace the 4-20mA conventional instrumentations with smart instrumentations communicating directly to the conventional controllers through the Foundation Fieldbus H1 bus, thus eliminating the need for I/O modules and the requirement for hard wiring from the instruments to the control system cabinets. The objective of the Foundation Fieldbus High Speed Ethernet network is to replace the proprietary control network and associated controllers by offering a standard linking device to connect H1 bus to the High Speed Ethernet network. The required control strategies within the controllers are distributed throughout the smart instrumentations. This concept will address the obsolescence challenges of the controllers and the control network provided that all required field instrumentations are smart. However, 80% of the field instrumentations are not compatible with Foundation Fieldbus H1 standard either discrete or conventional 4-20mA analog devices. This large part of the field instrumentation would still require the hard wiring as well as the proprietary controllers and associated I/O modules. [8]

Another attempt to address automation obsolescence challenges at the direct control functional level is implemented by Bedrock Automation. It focused on eliminating the root cause of automation obsolescence related to discontinuation of third party subcomponents due to obsolete electronic components. It owns almost all of the active semiconductor components required for the controller. The system is still proprietary and obsolescence challenges are not completely solved due to other root causes outside the control of the vendor such as difficult economic reception resulting in scaling down business or merging with other manufacturers. [16]

The latest attempt to address automation obsolescence challenges is to develop and open real-time system for all automation functional levels by Exxon Mobil and Lockheed Martin utilizing real-time service bus for network services. This study is in its infancy and aligned with the concept of developing vendor-independent automation solutions. [22]

2.7 Summary

For the past forty years, the development of process automation systems including programmable logic controllers has been evolving to raise productivity and enhance plant operation. Although each system has its own unique history and characteristics, they all share a basic workload objective of acquiring process data and controlling disparate machines in the plant to work in a cohesive fashion for improved safety, higher production rates, more efficient use of materials, and better consistency of product quality. Their fundamental architecture has advanced from large centralized system with all control hardware and input/output racks mounted in large cabinets located in the central control room to highly distributed systems.

Such systems typically have limited useful lives measured by the competitive advantage they deliver and the users have always struggled with determining their expected life spans. On top of this, large oil companies are making their revenue from their production, so shutting down the plant to replace the automation system due to premature obsolescence imposes a major challenge and definitely not a preferred solution.

In 2012, Automation Research Corporation Advisory Group performed a survey to determine the current state of the automation industry and best practices for managing the lifecycle of process automation systems from cradle to grave. There were 282 respondents from various parts of the process control industry including end users, suppliers, OEM manufacturers, and system integrators. The survey estimated the magnitude of the installed base assets of obsolete automation technology to be in the range of 65 billion US dollars. Extending the life cycle of these systems through incremental upgrades or migration paths would reduce the impact of obsolescence challenges to a limited extent. However, continuing to invest in proprietary solution is not economically attractive since similar obsolescence challenges will occur again at the end of the overall system lifecycle.

Responding to users' demand for a standard solution, automation vendors gradually started to incorporate COTS components in their automation solutions. However, the majority of the total automation solution including the controllers is based on proprietary hardware and software resulting in the requirement of major premature capital investments to sustain its operation throughout the life span of the controlled processing equipment.

From the literature review, the best practice for safeguarding against premature automation obsolescence is to avoid proprietary solution as much as possible by capitalizing on interoperable commercial-off-the-shelf, open source, and/or multi-supplier technologies.

The only technology to support multi supplier within a solution is the emerging heterogeneous computing based on real-time middleware technology. This technology provides the core capability of enabling standard commercial-off-the-shelf heterogeneous subcomponents from multi suppliers to cohesively work as one computing system. From the comprehensive literature review on middleware technologies, the standard real-time DDS middleware with QoS policies is identified as the core enabler for a comprehensive solution to solve the automation obsolescence challenges. These challenges include the requirement of in-kind spare part replacement of subcomponents, termination of technical support, inability to expand automation systems to meet new requirements, and inability to implement productivity enhancements. This technology has been utilized successfully for the past ten years in many mission-critical applications in the US military

CHAPTER 3

MOTIVATION

The significant advancements in the architecture of process automation systems in recent years has led to efficient increase in productivity and effective performance enhancements of operation plants. However, each system currently suffers from one or more of the following major disadvantages:

1. The current proprietary process automation systems typically have limited useful lives measured by the competitive advantage they deliver.
2. The requirement for overhauling the entire control application due to any critical changes in the control application or the process I/O signals.
3. There are inherent architectural constraints for achieving full utilization of the controllers' resources.
4. The PLC systems require expensive PIBs to house the marshaling and systems cabinets.
5. The application-centric architecture is not effective for integrating highly interacting heterogeneous process control applications across multiple network layers for exploiting processing facilities to achieve maximum yield of processing facilities.
6. The PLCs lack consistent and effective security protection at the control system level.
7. During PLC project execution, the actual system hardware including all I/O racks and required communication equipment are necessary to be available at the system staging area in the factory during testing, verification and factory acceptance test.

3.1 Obsolescence Challenges

Manufacturing industries are facing complex and persistent challenges over the past thirty years due to premature obsolescence of automation systems. The premature challenges include the requirement of in-kind spare part replacement of subcomponents, termination of technical support, inability to expand automation systems to meet new requirements, and inability to implement productivity enhancements. The root cause of the obsolescence challenges is having a single supplier of a proprietary automation system and one of the following scenarios occurs; natural product evolution, third party subcomponents discontinuation, merging with other manufacturers, quantum leap shift in technology, scaling down business, or filing for bankruptcy.

As the systems age; older technologies become obsolete resulting in frequent premature capital investments to sustain their operation. For example, when the controller becomes obsolete while the associated I/O system is still current and can be supported for the next 20 years, a premature capital intensive investment, about 75% of the total cost of ownership, is required for the replacement of the I/O system in order to replace the obsolete controller. Hence, retaining the current I/O system, when replacing the associated obsolete controller, is economically very attractive. However, this economic opportunity from a user perspective is not feasible due to either the lack of third party critical subcomponents or a new marketing strategy is adopted by the vendor to surpass its competitors and/or to increase its market share. Similar obsolescence challenges will be faced in fifteen to twenty years after the complete replacement of the existing obsolete control system. Therefore, the obsolescence challenges are complex and persistent cyclic problem mandating intensive capital investment to sustain the operation of the processing facilities throughout their life span.

3.2 Monolithic Architecture

The current PLCs are proprietary equipment based on a monolithic architecture, in which functionally distinguishable aspects such as the I/O system, the main control module, and the control application, are not architecturally separate components but are all interwoven. This is similar to the mainframe computer architecture. This architecture does not allow for changing the design of certain aspects of the controller easily without having to overhaul the entire control application or to buy another controller altogether. An example to demonstrate the disadvantage of the PLC monolithic architecture is the requirement for overhauling the entire control application due to some critical changes in the process I/O signals. Since the control application reference the actual physical address of the I/O signals inside the application including the rack number, module slot number, and the I/O channel number. For instance, the input for pump motor start control switch is referenced by “I3:4/2” where “I” referees to input module type, “3” referees to the rack number, “4” referees to the fourth module slot of the third rack, and “/2” referees to the second channel of the fourth I/O module. Obviously, changing the physical wiring of the input for pump motor start control switch from “I3:4/2” to another location “I5:1/6” due to a required voltage change in the motor start control circuit. The entire control application needs to be overhauled to reflect this change in every location “I3:4/2” is utilized.

3.3 Low Utilization

The PLC architecture is inherently constrained by one control application, maximum number of I/Os, maximum size of CPU memory in order to meet a minimum CPU scan time resolution. These constraints result in an overall low PLC systems utilization in the range of 30 to 50 percent. For example, design for maintainability best practice will not allow fitting two different control applications, each requiring only 30% of the controller capacity, in the same PLC. This is to avoid operation interruptions of a healthy control application while modifying, maintaining or troubleshooting the other control application residing in the same PLC. Therefore, two PLCs will be required with 30% utilization. On the other hand, suppose an existing PLC with a complex control application using 80% of the CPU memory and only 30% of the I/O system capacity. Expansion to this PLC requiring an additional of 30% of the CPU memory and only 10% of the I/O system capacity will not be feasible due to the constraint on the CPU memory. Therefore, an additional PLC will be required with a very low utilization to accommodate the new expansion. On the contrary, when the PLC CPU memory is less than 50% utilized and the PLC I/O capacity is completely utilized. An enhancement to the PLC control application requiring additional I/O signals will not be feasible due to the constraint of the maximum number of I/O and a split of the control application will be required into two PLCs resulting in low PLC utilization.

3.4 Expensive Initial Capital Cost

The PLC systems require PIBs with associated utilities to house the marshaling and systems cabinets. The PIBs are required to be blast proof and environmentally controlled structure to house the control systems equipment. These buildings require cabling to connect the hardwired I/O signals originating from the associated junction boxes all the way to the marshaling cabinets. This requirement for the PIBs increases the initial automation capital investment by approximately 20%.

3.5 Integration of Heterogeneous Applications

The overall system performance in furnishing dynamic process data updates depends highly on the level of integration among the heterogeneous process automation systems. The level of operation optimality and efficiency depends on the availability of such timely dynamic process data from all systems especially during abnormal situations. However, the PLCs have proprietary control communication network and lack seamless real-time integration with DCS, SCADA and/or the corporate enterprise resource planning system.

3.6 Security Protection

The PLCs lack consistent and effective security protection at the control system level. Some PLCs are provided with physical key lock to protect the memory and others do not have any protection. This security deficiency would result in islands of silos as a precaution for protecting the systems against malicious cyber-attacks.

3.7 Project Execution Schedule

During PLC project execution, the actual system hardware including all I/O racks and required communication equipment are necessary to be available at the system staging area in the factory during testing, verification and factory acceptance test. Holding the actual hardware at the factory and delaying the system installation phase for long time to validate the integrated control solution might result in a serious schedule delay on the overall project schedule hindering potential revenues of the main process applications.

CHAPTER 4

PROPOSED vCAP SOLUTION

4.1 vCAP Reference Model

To address the drawback of the monolithic mainframe architecture, modularity concept was utilized. Modular architecture is a design approach that subdivides a system into smaller parts called modules. These modules can be independently created and then used in different systems. Besides reduction in cost due to lesser customization and flexibility in design, modularity offers other benefits such as exclusion of obsolete modules and augmentation by merely plugging in different current modules. This idea allowed building computer systems with easily replaceable parts that use standardized interfaces and allowed upgrading or replacing obsolete aspects of the computer easily without having to buy another computer altogether. The main contribution of this dissertation is the development of an evergreen automation solution based on a modular architecture and vendor independent components that can last for the expected life span of the controlled process equipment. The life cycle of this automation solution can be managed and sustained using a replacement on failure strategy for all components. To achieve this objective, a virtual collaborative automation platform reference model is developed and applied in the design of the automation controllers to allow any upgrades or replacements of obsolete components without having to buy another automation controller altogether. This conceptual reference model characterizes the internal functions of an open and vendor independent automation controller by partitioning it into nine abstraction layers shown in Figure 20.

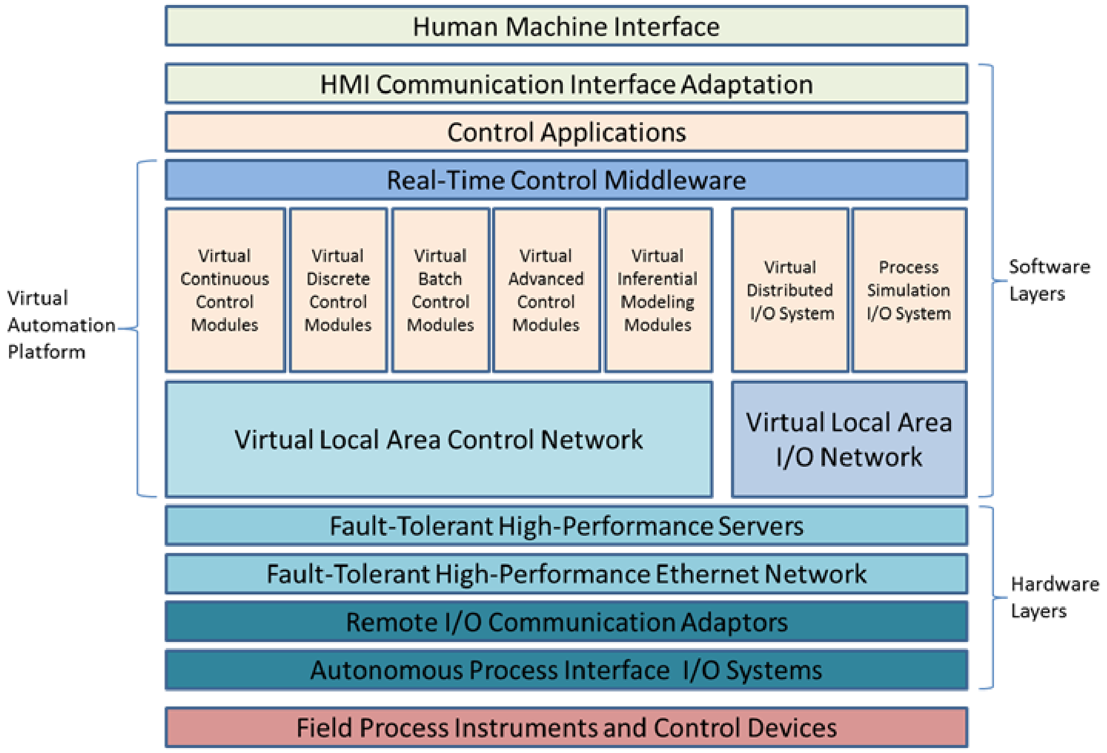

Human Machine Interface
HMI Communication Interface Adaptation
Control Applications
Real-Time Control Middleware
Virtual Continuous Control Modules
Virtual Discrete Control Modules
Virtual Batch Control Modules
Virtual Advanced Control Modules
Virtual Inferential Modeling Modules
Virtual Distributed I/O System
Process Simulation I/O System
Virtual Local Area Control Network
Virtual Local Area I/O Network
Fault-Tolerant High-Performance Servers
Fault-Tolerant High-Performance Ethernet Network
Remote I/O Communication Adaptors
Autonomous Process Interface I/O Systems
Field Process Instruments and Control Devices
Virtual Automation Platform
Software Layers
Hardware Layers

Figure 20 - Virtually Clustered Automation Platform Reference Model

The first four layers are physical hardware layers and the remaining are logical software layers. Each layer is independent of the other layers and the required modifications in each layer are independent of any in-progress modifications within the other layers. An abstraction layer is a way of hiding the implementation details of a particular set of functionality. It helps decoupling of the I/O system from the associated controller. The abstraction layers conceal all different I/O systems from the controllers by providing a uniform interface to all autonomous process interface I/O systems including COTS I/O systems, proprietary I/O legacy systems, and all I/O bus networks connected directly to field devices. The I/O bus networks are divided into two categories: device bus networks and process bus networks. Device bus networks interface with discrete devices such as limit switches and push buttons, while process bus networks interface with smart sensors such as control valves and process measurements including pressure, level, temperature, and flow. This abstraction will avoid the requirement to modify the control system when varying I/O hardware architecture. The same concept applies for changes in the main controller architecture do not require any modifications to the I/O hardware system. This provides a higher degree of decoupling when compared to the old conventional proprietary controller and its associated proprietary I/O system. The virtual collaborative automation platform consists of four main virtual components empowered by real-time control middleware; virtual control system, virtual I/O system, virtual I/O network, and virtual control network. This platform is primarily contained in three abstraction layers as shown in Figure 20. The following is a general description of all abstraction layers starting from the bottom, autonomous process interface I/O system layer.

4.1.1 Autonomous Process Interface I/O Systems Layer

The autonomous process interface systems layer corresponds to the standalone distributed I/O systems consisting of all required I/O hardware for connecting various filed process measurement and control devices including pressure, flow, level, and temperature instruments, motion sensors, position sensors, and final device elements such as motor operated gate valves and control valves to regulate the process pressure, flow, level or temperature. All field instruments are hardwired to the field junction boxes close to the associated process equipment. The autonomous I/O systems are not associated with the virtual controllers statically. Any I/O systems can be associated by one or more of virtual controllers dynamically as required. There are three types of process interface I/O systems included in this layer independent of the controllers and their manufacturers: COTS I/O system, existing proprietary I/O system, and the I/O bus networks.

4.1.1.1 Commercial Off-The-Shelf I/O System

The COTS I/O systems are currently used mainly for data acquisition and rarely for supervisory control through the web. There are three types of physical configurations, compact, fixed type modules, and mixed type modules. These I/O systems are enabled with either a single Ethernet communication port or redundant Ethernet communication ports. For those types support redundant Ethernet communication network, they can be daisy chained to form one logical autonomous process interface system with redundant Ethernet communication ports at both ends, first and last modules as shown in Figure 61.

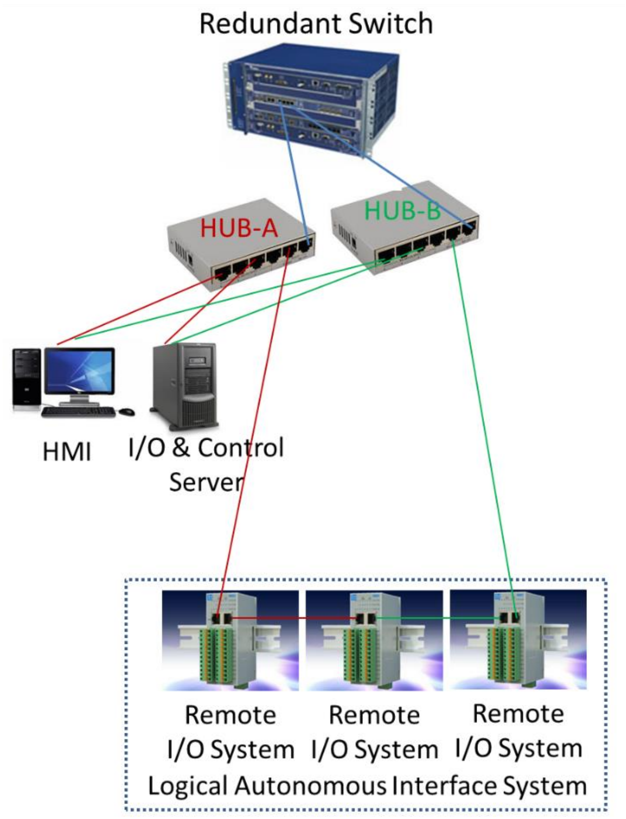


Figure 21 - Daisy Chained I/O Systems

The compact I/O system comes with mixed types of I/O and fixed number of I/O channels. For example, one type I/O system can support four 24 VDC discrete digital input channels, two 24 VDC discrete digital output channels, two 4-20 mA analog input channels, and one 4-20 mA analog output channel. The fixed type modular I/O system comes with specific types of I/O and fixed number of modules with fixed number of I/O channels. For example, one 24 VDC discrete input I/O system can support maximum of two modules with 8 channels each. The mixed type modular I/O system comes with any types of I/O modules, but fixed number of modules with different number of I/O channels. For example, a four-module backplane can support up to four I/O modules any types.

4.1.1.2 Existing Proprietary I/O System

The existing proprietary I/O systems are associated with proprietary PLCs and DCS controllers. In the PLC sphere, there are two types of physical configurations for the I/O system, fixed and modular. The CPU, power supply, and I/O system are all constructed as a single entity contained unit in the fixed PLCs. All the input and output screw terminals are built into the PLC package and are fixed, which cannot be moved. However, the modular PLC comes as separate pieces. All the parts of modular PLC are purchased separately, piece by piece. It’s like customizing, one may have 2 or 3 power supplies to choose from, a handful of different processors, many separate input modules, many separate output modules and a selection of assemblies, called racks, chassis, or baseplates to hold the pieces together. The I/O modules can be integral with the PLC or located remotely via a proprietary remote I/O communication network. The leading companies in the PLC business today are Siemens, ABB, Schneider (Modicon), Rockwell (Allen-Bradley), Mitsubishi, GE, Omron, Bosch Rexroth, Beckhoff, Fuji, and Toshiba. In a DCS, a hierarchy of controllers is connected by communications networks for command and monitoring. The I/O devices can be integral with the controller or located remotely via a proprietary field network. The leading companies in the DCS business today are ABB, Emerson, Honeywell, Invensys, Rockwell, Siemens, and Yokogawa.

4.1.1.3 I/O Bus Networks

In the sphere of I/O bus networks, several organizations such as the International Society of Automation and the European International Electronics Committee are working together for establishing network and protocol standards. In the process bus area, two main organizations are working toward establishing protocol standards: the FOUNDATION Fieldbus and PROFIBUS. In the device bus network, several bus protocol standards exist with Seriplex, AS-Interface, and InterBus. [51]

4.1.2 Remote Input/Output Communication Adaptors Layer

The purpose of the I/O communication adaptation layer is to ensure that all I/O systems in the layer underneath it are capable of interfacing to redundant Ethernet communication links. For the COTS I/O system, only the I/O system that supports a single Ethernet communication network requires an additional Ethernet hub to accept redundant Ethernet communication links coming from the Fault-Tolerant Ethernet network and one communication link connected to the I/O system as shown in Figure 22.

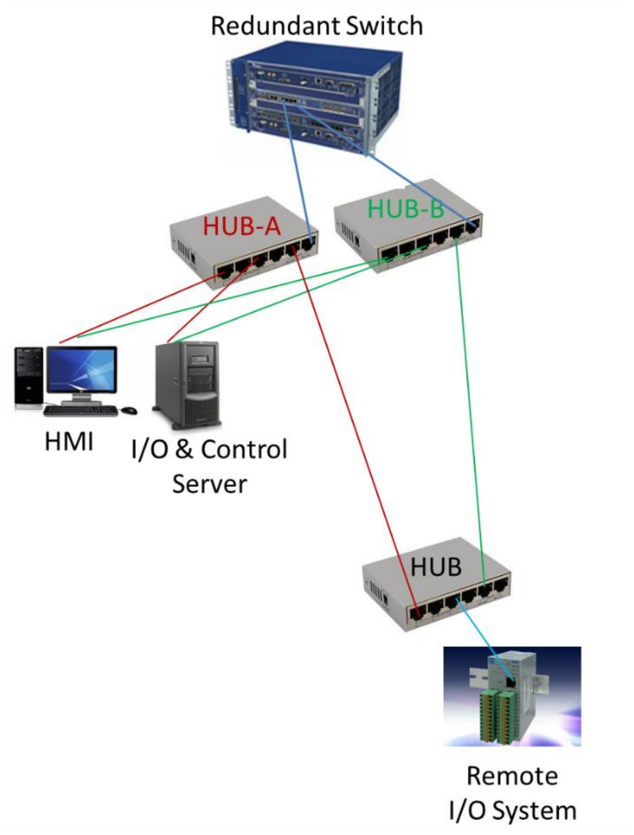


Figure 22 - Adaptation Hub for COTS I/O System with Single Ethernet Port

For the existing proprietary I/O system, there is a need to develop a customized communication adaptor equipped with redundant Ethernet communication ports to communicate with the I/O server for sending input status and receiving output commands. For the I/O bus networks, standard communication adaptors are required to provide redundant Ethernet communication ports as well as a connection to the specific I/O bus network. For example, there is a standard Ethernet/Foundation Fieldbus convertor called linking device and a standard Ethernet/ PROFBUS convertor called Anybus. This adaptation layer abstracts the complexity of the communication interface to all I/O systems by transforming them to standard Ethernet communication links.

4.1.3 Fault-Tolerant High-Performance Ethernet Network Layer

The fault-tolerant high-performance Ethernet network layer defines the physical local area network architecture for interconnecting the I/O systems and the required fault-tolerant servers as well as the required HMI PCs. The local area network is based on Ethernet with physical hierarchical star topology. It consists of two levels, backbone and device levels. The backbone level consists of two redundant high-performance switches interconnected using pair of communication links. The device level includes multiple dual hubs connected to the associated redundant switch. Each end device such as the autonomous process interface I/O system, server, or PC is connected to the associated dual hubs by two links, one link per hub as shown in Figure 23.

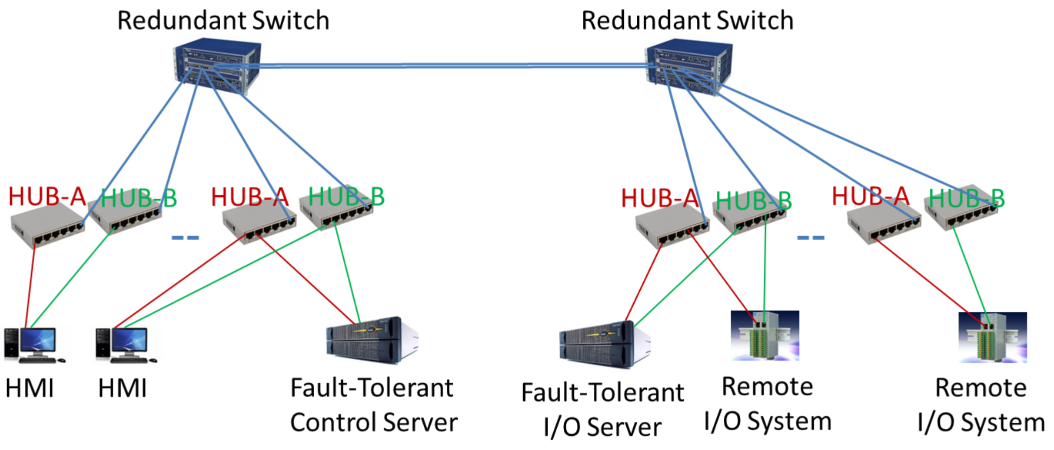


Figure 23 - Fault-Tolerant High-Performance Ethernet Network

The location of the redundant switches is the server room within the Central Control Room. The temperature and humidity in the server room have a direct relationship to the proper functioning of the installed network switches. Also, the installation of the network switches has a direct relationship to the proper protection from excessive vibration and shock. The location of the pair of hubs for connecting the servers is the server room, for connecting the HMI PCs is the control room, and for connecting the remote I/O systems is the master field junction box. The junction boxes in the field are scattered and are located close to the associated process equipment for connecting the associated instruments. They form multiple clusters. Each cluster of the field junction boxes requires a pair of hubs to connect them to the Ethernet communication network. The pair of hubs is located in the master field junction box. The master junction box is identified based on the minimum total distance when connecting all field junction boxes within a cluster to a selected junction box. The temperature, humidity, vibration and shock are very crucial when installing the pairs of hubs in the master field junction boxes. The hubs are required to withstand harsh environments, -40°C to +80°C at 100% humidity, without the need for any environmental control. [52]

4.1.4 Fault-Tolerant High-Performance Servers Layer

For any enterprise facing challenges in managing data effectively, one of the dominant technology trends of the day is virtualization. The more servers are virtualized, the fewer physical machines are needed. The benefits of virtualization include reduced total cost of ownership, improved and centralized manageability, optimal resource utilization, less power consumption, and reduced footprint. The single biggest threat that consolidated servers face is that if just one consolidated server goes down, many applications will be impacted. To avoid this scenario, the concept of Fault Tolerance becomes very crucial. Fault-tolerance describes the computer system design so that, in the event that a component fails, a backup component or procedure can immediately take its place with no loss of service. Fault tolerance can be provided with software, or embedded in hardware, or provided by some combination. The fault-tolerant high-performance servers' layer defines the requirements for the required computing platforms to emulate the distributed hardwired I/O systems and associated controllers. Fault-tolerant computing is the ability to provide the world's most demanding workloads with 99.999% system uptime or better, zero failover time and no data loss. The fault-tolerant server is required to guarantee zero downtime and maximum performance, prevent data loss and corruption including in-flight data, enable virtualization for applications, support real-time and high-volume workloads with ease, and accommodate network and transaction growth. [54]

4.1.5 Virtual Local Area Networks Layer

A virtual local area network can be created by partitioning a physical local area network into multiple mutually isolated logical LANs using a virtual LAN ID. A virtual LAN has the same attributes as a physical local area network, but it allows for end stations to be grouped together more easily even if they are not on the same network switch. Virtual LAN membership can be configured through software instead of physically relocating devices or connections. The purpose of the virtual local area networks layer is to define the requirements of creating two virtual local area networks, one dedicated for distributed I/O processing and another one dedicated for control applications. The physical fault-tolerant high-performance Ethernet network is partitioned into two virtual LANs, virtual local area I/O network and virtual local area control network. IEEE 802.1Q is the networking standard that supports virtual LANs on an Ethernet network.

4.1.6 Virtual Controllers Layer

The goal of virtualization is to centralize administrative tasks while improving scalability, security, and overall hardware-resource utilization. The virtual controllers' layer defines the main virtual computing engines required for a comprehensive virtual distributed control system environment. Minimum of two fault-tolerant high-performance servers are required, one for I/O system processing and another one for control processing. The I/O system server is virtualized into multiple operation areas. For example, in a GOSP application, the I/O system server is fully virtualized into three virtual I/O servers for crude oil handling, gas compression, and utility. The control server is fully virtualized into ten virtual discrete control modules, one virtual continuous control module, one virtual advanced control module, and one virtual inferential model prediction. The crude oil handling area requires four virtual discrete control modules. The gas compression area requires four virtual discrete control modules. The utility area requires two virtual discrete control modules. There is also a need for one conventional server for the offline process simulation. The virtual control applications and I/O systems can be distributed into multiple physical fault-tolerant control and I/O servers to sustain the applications performance. The interaction among the virtual and conventional servers is processed through the real-time control middleware. [57]

4.1.7 Real-Time Control Middleware Layer

The real-time control middleware layer is the heart of the virtual automation controller architecture and is based on the data distribution service middleware technology. This middleware is the most advanced and efficient standard-based technology for real-time data distribution and serves as a glue to connect the virtual distributed I/O systems, the virtual controllers, the HMIs, and the offline process simulation I/O system. This technology allows for decoupling the I/O systems from the controllers. Therefore, allowing dynamic soft association between the I/O system and the controller rather than a static and physical association in the conventional proprietary controllers where the I/O system is a fundamental part of the controller body. For example when utilizing conventional proprietary controllers, if new additional critical process measurements for certain process equipment are required and they are very far from the associated controller and very close to a nearby controller handling different processes and functionalities, these new measurements cannot be hardwired to the nearby controller in order to optimize the capital cost. However, these new measurements must be hardwired to the associated controller where the control application resides regardless of the associated capital cost. On the other hand, the usage of real-time data distribution service middleware technology provides the flexibility of decoupling the I/O systems from the virtual controllers. These new measurements can be hardwired to the nearby I/O system in order to optimize the capital cost and the association of the new measurements and the applicable virtual controller is done in soft through the middleware publish/subscribe relationship process. Another example for the benefit of decoupling the I/O system from the controller is switching between the simulation environment and the real hardware environment is completely transparent to the control logic because of the I/O system hardware abstraction.

4.1.8 Control Applications Layer

The control applications layer defines the application programming environment for all types of virtual controllers. The control applications use the IEC 61131-3 standards-based programming languages including function block diagram, ladder logic diagram, structured text, instruction list, and sequential function chart.

4.1.9 HMI Communication Interface Adaptation Layer

The HMI communication interface adaptation layer defines the required gateway convertor from Ethernet-based network to the proprietary control network of the DCS or SCADA systems.

4.2 vCAP Design Evolution

The architecture of the proposed virtual collaborative automation platform has evolved from the advancement in five standard state of the art technologies: fault-tolerant computing, industrial fault-tolerant Ethernet networking, virtualization of computing and networking, real-time middleware, and autonomous I/O systems. The overall control system architecture based on vCAP is shown in Figure 24. This architecture consists of six primary components; virtual control system, virtual I/O system, HMI data servers, real-time control middleware, virtual communication networks, and autonomous process interface I/O systems distributed among junction boxes. [79]

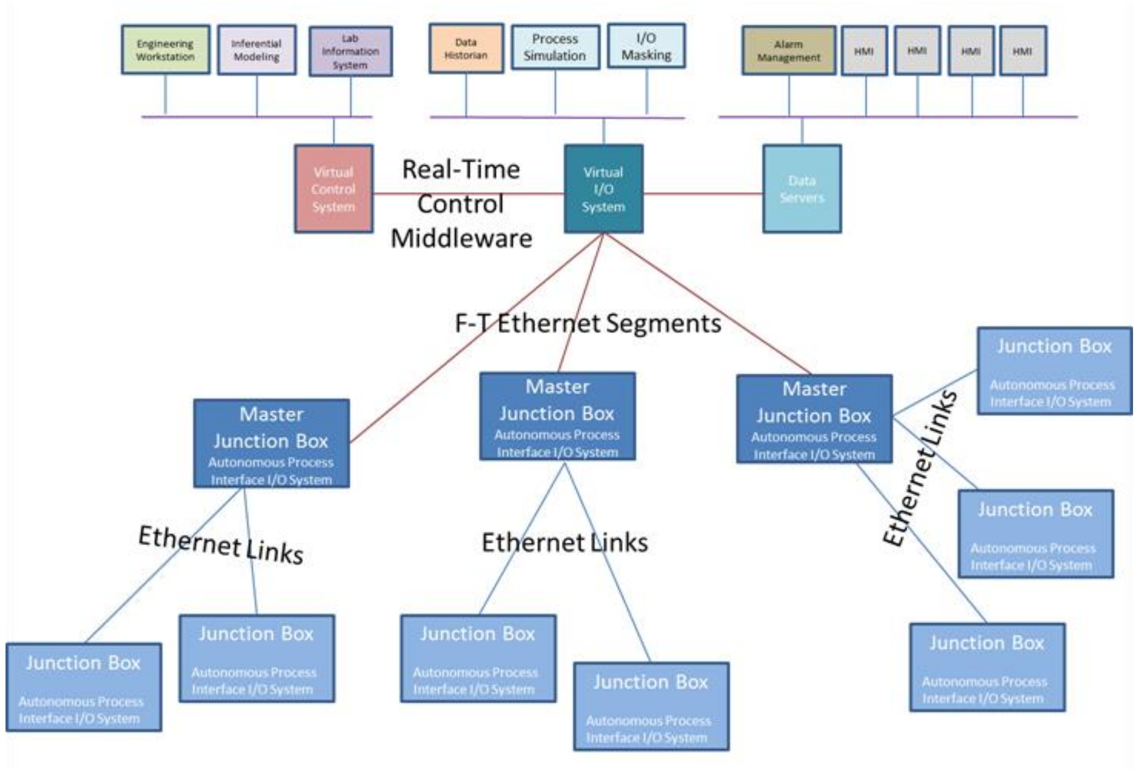

Engineering Workstation
Inferential Modeling
Lab Information System
Data Historian
Process Simulation
I/O Masking
Alarm Management
HMI
HMI
HMI
HMI
Virtual Control System
Real-Time Control Middleware
Virtual I/O System
Data Servers
F-T Ethernet Segments
Master Junction Box Autonomous Process Interface I/O System
Master Junction Box Autonomous Process Interface I/O System
Master Junction Box Autonomous Process Interface I/O System
Ethernet Links
Ethernet Links
Ethernet Links
Junction Box Autonomous Process Interface I/O System
Junction Box Autonomous Process Interface I/O System
Junction Box Autonomous Process Interface I/O System
Junction Box Autonomous Process Interface I/O System
Junction Box Autonomous Process Interface I/O System
Junction Box Autonomous Process Interface I/O System
Junction Box Autonomous Process Interface I/O System

Figure 24 - Overall Control System Architecture based on vCAP

4.2.1 Virtual Computing Platform Evolution

4.2.1.1 Virtual Computing Servers

Hardware virtualization refers to the creation of a virtual machine. The host machine is the actual machine on which the virtualization takes place, and the guest machine is the virtual machine. The software or firmware that creates a virtual machine on the host hardware is called a hypervisor or virtual machine manager. [9]

The goal of virtualization is to centralize administrative tasks while improving scalability and overall hardware-resource utilization. With virtualization, several operating systems can be run in parallel on a single central processing unit. This parallelism tends to reduce overhead costs and differs from multitasking, which involves running several programs on the same operating system. Instead of relying on the current model of “one controller per one control application” that leads to underutilized resources, virtual controllers are dynamically applied to meet business needs without any excess fat. However, in abnormal situations where a single hardware failure may cause many control applications running in the same hardware may fail together resulting in drastic operation interruption. Although virtualization provides better hardware utilization, the virtualization infrastructure including the hypervisor and privileged virtual machines remain vulnerable to hardware errors. Therefore, as a price for the benefits of adopting virtualization to improve hardware utilization and management, the hardware platforms’ operational availability need to be addressed. [10]

4.2.1.2 Higher Availability Virtual Computing Servers

There are a number of conventional approaches to assuring server availability. For less expensive servers, one common method is to cluster multiple servers with a master server monitoring them and software to switch slave servers in the event of hardware failure. When a failure occurs, the clustering software immediately starts the application on the standby system without requiring administrative intervention. [17]

Another method for assuring server availability is to use fault-tolerant computing. Fault-tolerant computing is the art and science of building computing systems that continue to operate satisfactorily in the presence of faults. The majority of fault-tolerant designs have been directed toward building computers that automatically recover from random faults occurring in hardware components. The techniques employed to do this generally involve partitioning a computing system into modules that act as fault-containment regions. Each module is backed up with protective redundancy so that, if the module fails, others can assume its function. Special mechanisms are added to detect errors and implement recovery. [18]

vCAP architecture is based on utilizing multiple fully virtualized hardware-based fault-tolerant servers for the control applications and multiple fully virtualized hardware-based fault-tolerant servers for the I/O database.

4.2.2 Real-Time Control Middleware Evolution

DDS for real-time systems aims at enabling scalable, real-time, dependable, high-performance, and interoperable data exchange system among publishers and subscribers. The DDS specification describes two levels of interfaces: a lower data-centric publish-subscribe level that is targeted towards the efficient delivery of the proper information to the proper recipients, and an optional higher data local reconstruction layer level, which allows for a simple integration of DDS into the application layer. Both commercial and open-source implementations of DDS are available. [59]

DDS allows the user to specify quality of service parameters to configure discovery and behavior mechanisms up-front. These QoS polices are essential for real-time systems in order to manage the middleware behavior and to sustain and guarantee the overall system performance.

4.2.2.1 DDS Quality of Service Polices For Real-Time Systems

DDS QoS policies for real-time systems can be used to control and optimize network as well as computing resource to ensure the right information is delivered to the right subscriber at the right time. They are classified into five categories: data availability, data delivery, data timeliness, resources, and configuration. [62]

4.2.2.1.1 Data Availability

The durability QoS policy controls the data availability with respect to late joining publishers and subscribers.

The lifespan QoS policy is to avoid delivering “stale” data to the application.

The history QoS policy controls whether the DDS should deliver only the most recent value, attempt to deliver all intermediate values, or do something in between.

4.2.2.1.2 Data Delivery

The presentation QoS policy specifies how the samples representing changes to data instances are presented to the subscriber.

The reliability QoS policy indicates the level of guarantee offered by the DDS in delivering data to subscribers.

The partition QoS policy allows the introduction of a logical partition concept inside the ‘physical’ partition induced by a domain.

The destination order QoS policy controls how each subscriber resolves the final value of a data instance that is written by multiple publishers running on different nodes.

The ownership QoS policy specifies whether it is allowed for multiple publishers to write the same instance of the data and if so, how these modifications should be arbitrated.

The ownership strength QoS policy specifies the value of the “strength” used to arbitrate among publishers that attempt to modify the same data instance.

4.2.2.1.3 Data Timeliness

The deadline QoS policy allows defining the maximum inter-arrival time between data samples.

The latency budget QoS policy specifies the maximum acceptable delay from the time the data is written until the data is inserted in the receiver's application-cache.

The transport priority QoS policy is a hint to the infrastructure as to how to set the priority of the underlying transport used to send the data.

4.2.2.1.4 Resources

The time based filter QoS policy allows a subscriber to indicate that it does not necessarily want to see all values of each instance published under the topic.

The resource limits QoS policy controls the resources that the service can use in order to meet the requirements imposed by the application and other QoS settings.

4.2.2.1.5 Configuration

The user data QoS policy is to allow the application to attach additional information to the created entity objects such that when a remote application discovers their existence it can access that information and use it for its own purposes.

The topic data QoS policy is to allow the application to attach additional information to the created topic such that when a remote application discovers their existence it can examine the information and use it in an application-defined way.

The group data QoS policy is to allow the application to attach additional information to the created publisher or subscriber.

4.2.2.2 DDS Interoperability Wire Protocol

With the increasing adoption of DDS in large distributed systems, it was desirable to define a standard “wire protocol” that allows DDS implementations from multiple vendors to interoperate. Hence, OMG developed the real-time publish-subscribe wire protocol DDS interoperability wire protocol specification for defining an interoperability protocol to ensure that information published on a topic using one vendor's DDS implementation is consumable by one or more subscribers using the same or different vendor's DDS implementations. The “DDS wire protocol” is capable of taking advantage of the QoS settings configurable by DDS to optimize its use of the underlying transport capabilities. In particular, it is capable of exploiting the multicast, best-effort, and connectionless nature of many of the DDS QoS settings. [63]

4.2.3 Fault-Tolerant Control Network Evolution

Basic Ethernet is a broadcast topology that may be structured as a physical bus topology or a hub, physical star topology with a logical bus structure, as shown in Figure 25.

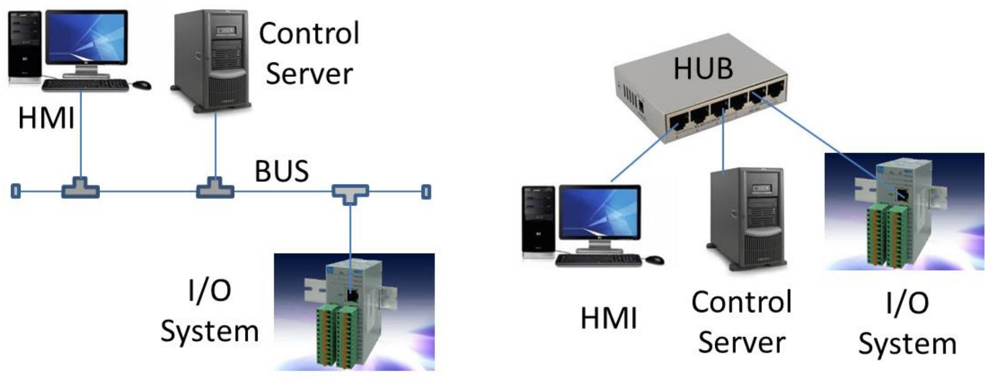


Figure 25 - Bus-Based Ethernet Network

The advantages of bus topology include: easy to connect a node to a linear bus, requires less cable length than a star topology, and it works well for small networks. The disadvantages of bus topology include: entire network shuts down if there is a break in the main cable, terminators are required at both ends of the backbone cable, difficult to identify the problem if the entire network shuts down, not meant to be used as a stand-alone solution in a large network, it is slow when more nodes are added into the network, and if a main cable is damaged then network will fail or be split into two networks. [12]

In a dynamic redundant Ethernet network, the redundancy is not actively participating in the control. A switchover logic mechanism decides to insert redundancy and put it to work. It is an analogy of having a spare tire in the car. This paradigm allows sharing redundancy and load, implementing partial redundancy, reducing failure rate of redundancy, and reducing common mode of errors. But, the switchover mechanism takes time to recover. For example, in case of a link failure in the dynamic redundant network shown in Figure 26, the switchover mechanism routes the traffic over another port in order to avoid using the failed link. [78]

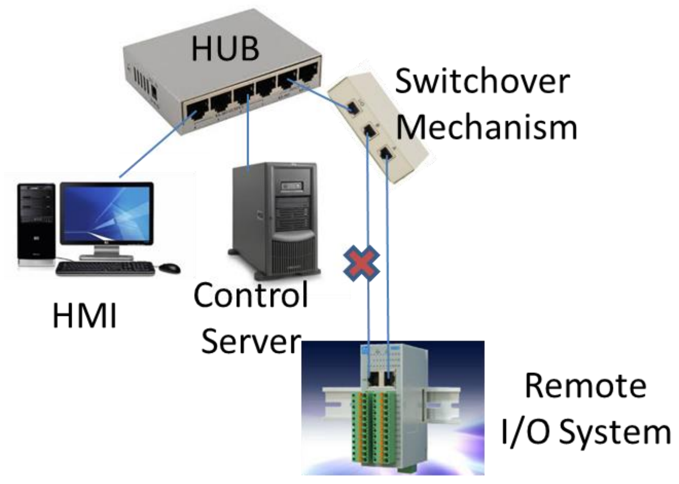


Figure 26 - Dynamic Redundant Ethernet Network

The switchover process normally takes time for detecting the link failure and reconfiguring the switchover mechanism to select the healthy link. The singly attached nodes including the switchover mechanism and the associated switch and links are assumed to be trusted components. This example demonstrates the implementation of partial redundancy where the redundancy is applied only for connecting long distance remote nodes in order to protect against cable failures. The redundant links usually take different physical routes to overcome accidental errors in ground civil construction.

In a static redundant Ethernet network, the redundancy is participating in the control. The user chooses the trusted working unit. It is an analogy of having double tires in trucks. This paradigm allows providing bumpless switchover, continuously exercising redundancy and increasing fault detection coverage, and providing fail-safe behavior. But, it costs total duplication. For example, in case of a link failure in the static redundant network shown in Figure 27, the doubled attached nodes work with the remaining channel instantaneously without any needs for changing the routing tables. This is because there are two disjoint paths from each source to any destinations. Each node has redundant interface ports for connecting to both networks and assumed to be trusted component. Any failures in one network will cause the whole network to fail and gets disabled silently; however, the other network will continue to be utilized for communication among all connected nodes. This configuration can tolerate any single failures in the nodes, hubs, links or interface ports. Therefore, a single failure in each network will cause a complete system communication failure. Multiple failures in one network can be tolerated as long as the other network is in perfect condition. This configuration is easy and efficient, but expensive and not effective.

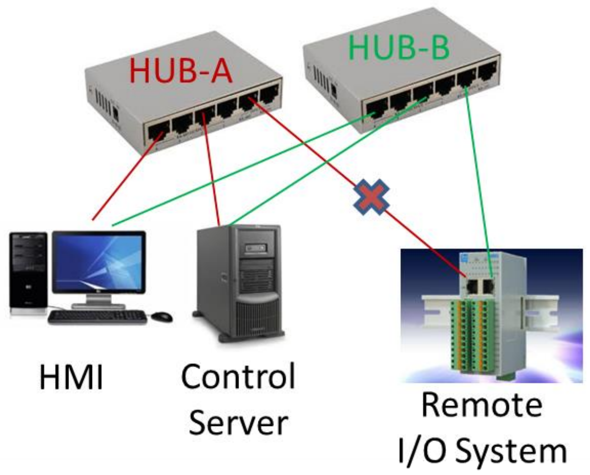


Figure 27 - Static Redundant Network

If the redundancy is in the network only and the nodes are singly attached as shown in Figure 28, the protection will be against network component failures only. The nodes and associated links are trusted components.

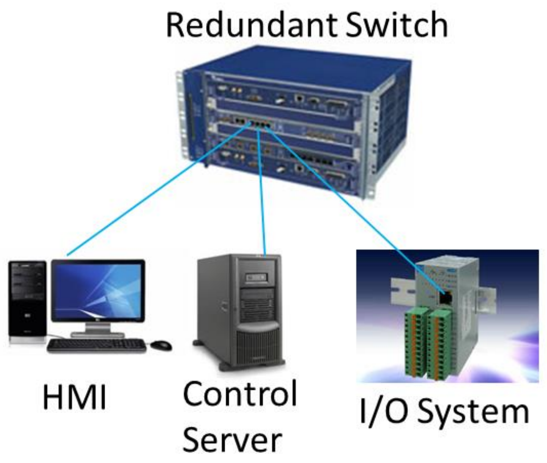


Figure 28 - Redundancy in the Network Only

If each node is functionally redundant and consists of duplicate hardware, but singly attached to the redundant network, the protection will be against all failures including node or network failures as well as the network adapters as shown in Figure 29.

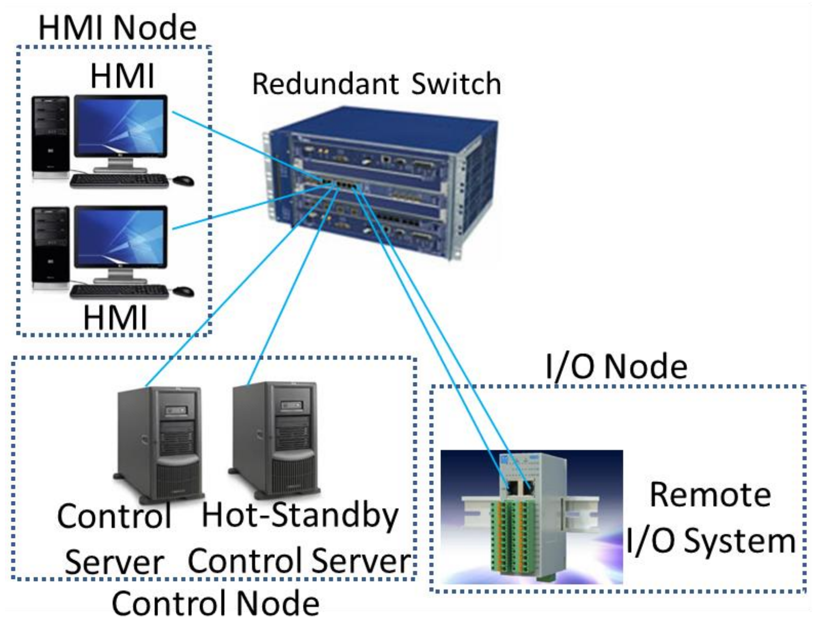


Figure 29 - Redundant Network with Duplicate Nodes of Singly Attached Modules

There are two redundant paths from any source functionally redundant nodes to any destination functionally redundant nodes, but very costly.

In a hybrid redundant Ethernet network, the redundancy is divided into two levels: backbone level based on dynamic redundancy using a redundant Ethernet switch, and device level based on static redundancy using pair of hubs as shown in Figure 30. It combines the advantages of both static and dynamic models and improves the network availability in cost effective manner. In this configuration, there are 2 disjoint paths and 2 redundant paths between any source and destination nodes without the need for node duplication. The 2 disjoint paths do not go through the redundant switch. However, the other 2 redundant paths go through the redundant switch. The protection is against network components failure as well as the network adapter failures. However, the nodes are trusted elements.

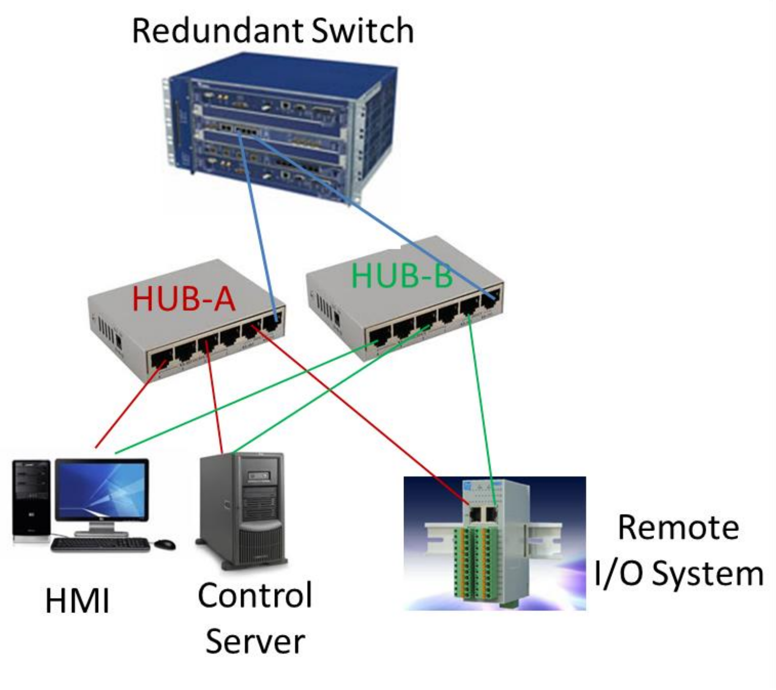


Figure 30 - Hybrid Redundant Ethernet Network with Redundant Node Links

To improve the availability of the nodes, fault-tolerant nodes with redundant interface links are used for critical applications. For non-critical applications, duplicate hardware for the node is used with balance loading. For example, Figure 31 shows the same network as the one in Figure 30; however, the control server is replaced with fault-tolerant control server and added an additional HMI. The control application is critical and the fault-tolerant control server will ensure bumpless transfer when failures occur; however, there is no load balancing. On the other hand, the load of the HMI application is distributed between both HMI nodes. When one HMI fails, the other HMI will assume the full load, in degraded mode.

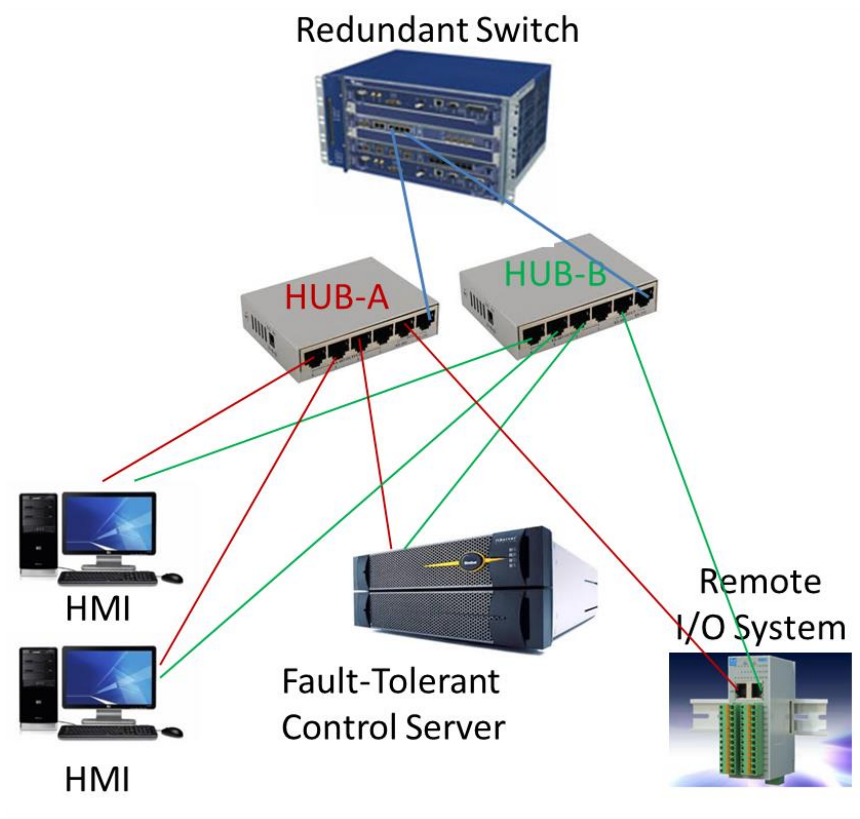


Figure 31 - F-T Ethernet Network with Redundant Nodes and Links

To improve the network performance, it can be duplicated as shown in Figure 32, by aggregating the nodes with high interaction under a redundant switch. For example, the interaction between the I/O server and the I/O systems is very high and are located under the second redundant switch. The redundant switches are interconnected by redundant links. In this configuration, there are 2 disjoint paths between any source nodes and destination nodes within the pair of hubs. There are also 4 redundant paths between any source nodes and destination nodes within one redundant switch. Finally, there are 8 redundant paths between any source nodes under one redundant switch and any destination nodes under the other redundant switch.

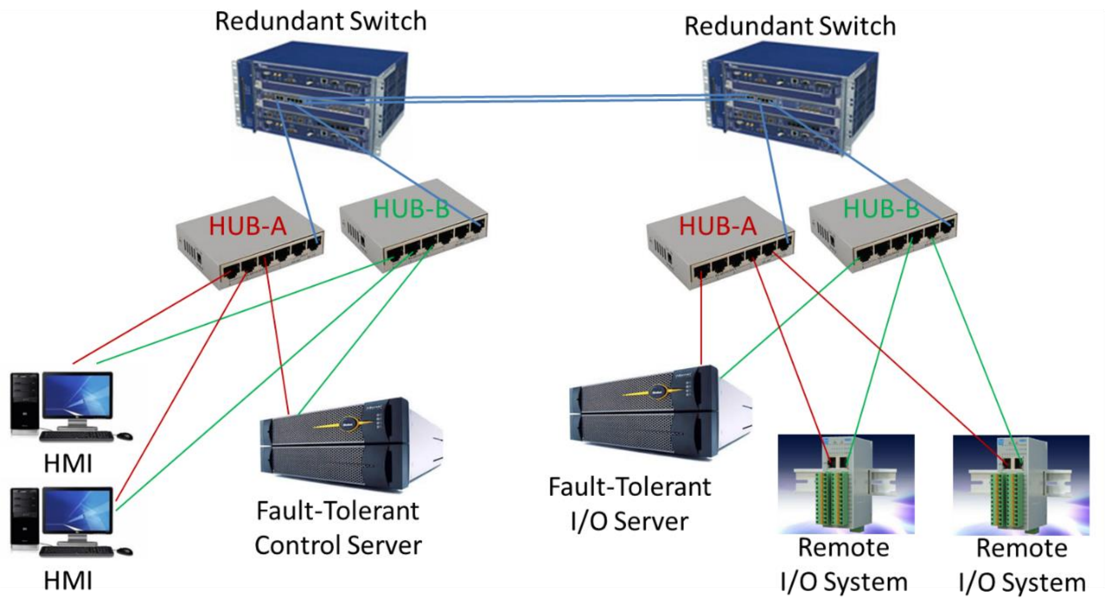


Figure 32 - Expanded F-T Ethernet Network with Redundant Nodes and Links

4.3 vCAP Architectures

4.3.1 Basic vCAP Architecture

The basic vCAP architecture consists of one high-performance server, to be located in the server room of CCR, two Ethernet switches: one switch serves as the backbone switch located in the server room and the other one serves as the linking I/O switch located in the master junction box, and one or more autonomous process interface I/O systems distributed in the smart junction boxes as shown in Figure 33. The high-performance server can be fully virtualized into many virtual servers: some for the virtual controllers and the other ones for the virtual I/O systems. Figure 85 includes only one virtual controller and one virtual I/O system.

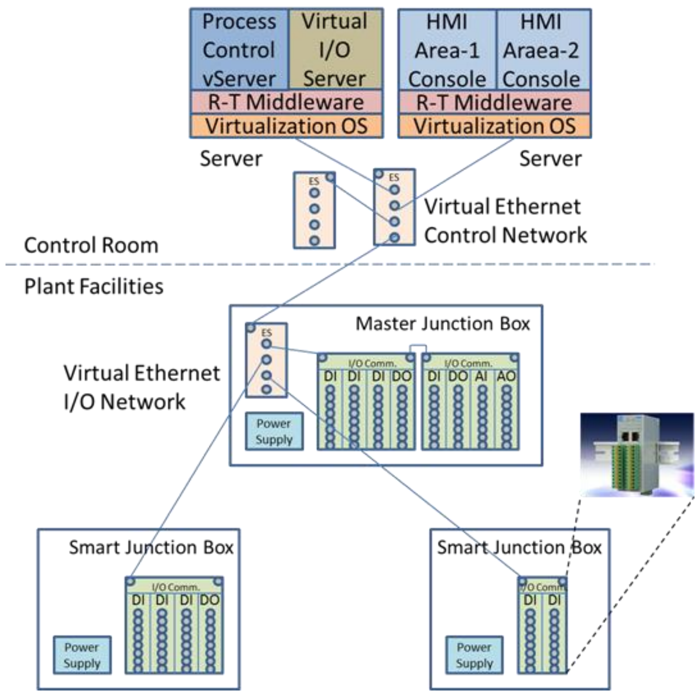


Figure 33 - Basic vCAP Architecture

Attached to the backbone Ethernet switch are the high-performance server, the HMI server, and the linking I/O Ethernet switch. The link extended from the backbone Ethernet switch to master junction box is called I/O segment. The I/O system is scalable and can handle multiple I/O segments by expanding the backbone Ethernet switch if required. The process interface I/O systems located at the smart junction boxes are connected to the linking I/O Ethernet switch located in the master junction box. The Ethernet network is virtualized into two virtual local area networks: one for control and the other one for I/O processing. The virtual local area control network includes the virtual control server, the virtual I/O server, and the HMI server. The virtual local area I/O network includes the virtual I/O server and the autonomous process interface I/O systems. The virtual I/O server is a common node in both virtual local area networks. The communications among the nodes attached to the virtual local area control network are accomplished through the real-time publish/subscribe messaging middleware. The communications between the master nodes, the virtual I/O systems, and the slave nodes, autonomous process interface I/O systems, attached to the local area I/O network are accomplished using report by exception with integrity poll communication protocol. The virtual I/O server creates topics, one topic per I/O channel named by the device tag. The virtual I/O server updates the status of the device tags by communicating to the autonomous process interface I/O systems using report by exception with integrity poll communication protocol. This update process is accomplished independently from the virtual control servers. The junction box becomes smart in vCAP architecture since it includes I/O system that can act as a part-time master to report by exception a status update of the associated I/O channels. The location selection criteria for the master junction boxes are identical to the selection criteria for the locations of the PIBs. This architecture is equivalent to the conventional modular PLC architecture with remote I/O expansions illustrated in Figure 30. However, the vCAP can handle more than one control applications by increasing the number of virtual controllers and I/O systems in the high-performance server. For this reason, improving the reliability and availability of the high-performance server is crucial. This can be accomplished using software-based fault-tolerance mechanism or hardware-based fault-tolerance mechanism.

4.3.2 Software-Based Fault-Tolerant vCAP Architecture

VMware vSphere fault tolerance provides continuous availability for applications in the event of server failures by creating a live shadow instance of a virtual machine that is always up-to-date with the primary virtual machine. In the event of a hardware outage, vSphere automatically triggers failover ensuring zero downtime and preventing data loss. This fault-tolerance scheme is applied for both the virtual controllers and the virtual I/O systems as shown in Figure 34. The HMI server is replaced with two PC-based HMI consoles that can handle all areas of operation. In this architecture example, the virtual local area control network includes one fault-tolerant virtual controller, one fault-tolerant virtual I/O system, and two HMI PCs. The virtual local area I/O network includes one fault-tolerant virtual I/O system and the autonomous process interface I/O systems. This architecture is equivalent to conventional redundant modular PLC architecture shown in Figure 31. However, the vCAP can handle more than one control applications by increasing the number of virtual controllers and I/O systems in the high-performance servers. Economy of scale is an important aspect of the fault-tolerant vCAP architecture; one hardware redundancy for the high-performance server utilized for multiple virtual controllers compared to individual hardware redundancy for each conventional redundant PLC system.

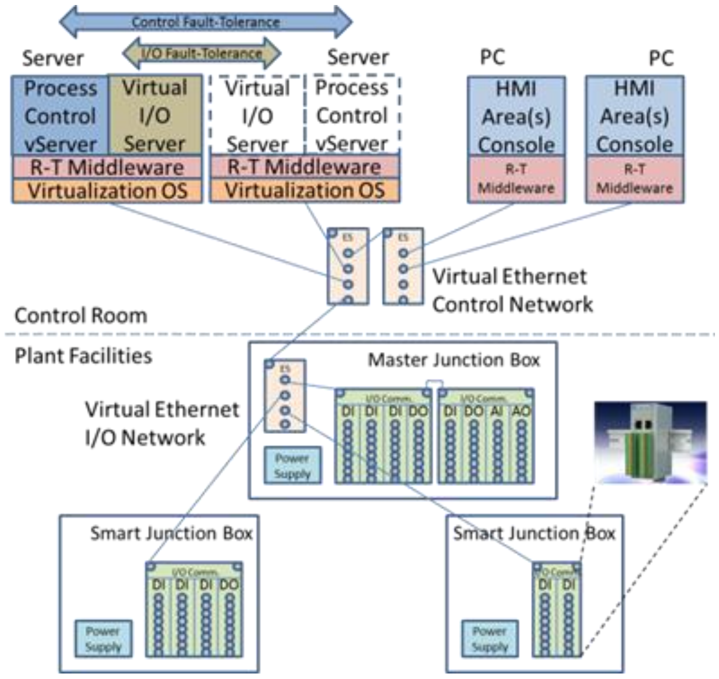


Figure 34 - Software-Based Fault-Tolerant vCAP Architecture

4.3.3 Hardware-Based Fault-Tolerant vCAP Architecture

Several vendors provide hardware-based fault-tolerant servers. These servers have completely duplexed hardware in a single chassis. The single operating system and application run in lockstep on both sets of hardware, and if there is an issue, the parallel hardware component keeps on running without any downtime, or failover. This fault-tolerance scheme is applied for both the virtual controllers and the virtual I/O systems as shown in Figure 35. In this architectural example, this fault-tolerance scheme is also applied for the HMI consoles. This architecture is equivalent to conventional redundant DCS controller with simplex I/O modules.

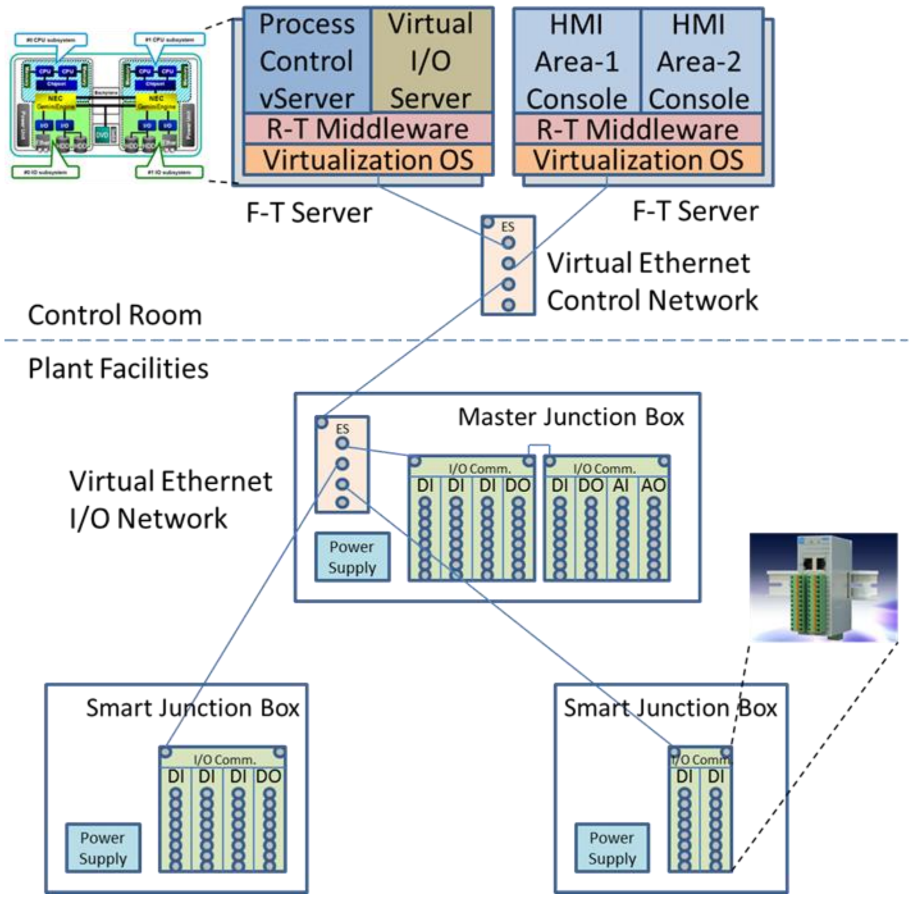


Figure 35 - Hardware-Based Fault-Tolerant vCAP Architecture

4.3.4 Fault-Tolerant vCAP with Parallel Network Architecture

To improve the reliability of the hardware-based fault-tolerant vCAP architecture, the Ethernet network is replaced with parallel Ethernet network where a complete duplicate of the hardware components and communication links are working in parallel as shown in Figure 36.

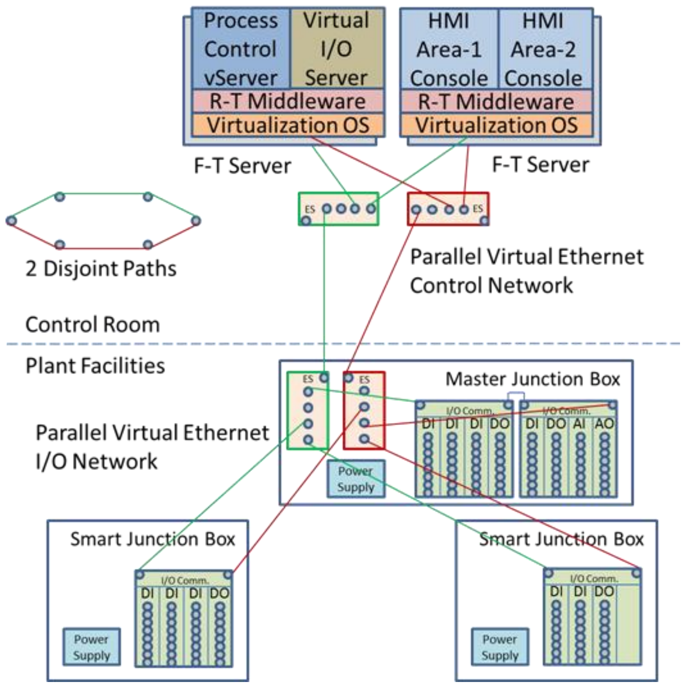


Figure 36 - Fault-Tolerant vCAP with Parallel Network Architecture

This architecture provides two disjoint paths between the fault-tolerant servers and between the fault-tolerant virtual I/O system and any autonomous process interface I/O systems. The standard parallel redundancy protocol is used to allow the parallel Ethernet network overcoming any single network failures without affecting the data transmission. It is independent of the protocols used in the virtual local area control and I/O networks and provides seamless failover. These I/O systems are enabled with either a single Ethernet communication port or redundant Ethernet communication ports. The I/O systems with single Ethernet communication network require an additional Ethernet hub to accept redundant Ethernet communication links coming from the parallel Ethernet network and one communication link connected to the I/O system as shown in Figure 22. For multiple autonomous process interface I/O systems that support redundant Ethernet communication network within one smart junction box, they can be daisy chained to form one logical autonomous process interface I/O system with redundant Ethernet communication ports at both ends, first and last modules as shown in Figure 21. This architecture is equivalent to conventional redundant DCS controller with redundant I/O communication, but with simplex I/O modules. Conceptually, the reliability of the fault-tolerant vCAP system with parallel network architecture is higher than the conventional redundant modular PLC system due to the redundancy of the I/O communication links. However, any failures in one network leg will cause the whole network leg to fail and gets disabled silently. The other network leg will continue to be utilized for communication among all connected nodes. This configuration can tolerate any single failures in the nodes, hubs, links or interface ports. Therefore, a single failure in each network will cause a complete system communication failure. Multiple failures in one network leg can be tolerated as long as the other network leg is in perfect condition.

4.3.5 Fault-Tolerant vCAP with Fault-Tolerant Network Architecture

To improve the reliability of the hardware-based fault-tolerant vCAP architecture, the parallel Ethernet network is replaced with fault-tolerant Ethernet network where any single failures can be tolerated as well as multiple failures provided that there is at least one redundant path available between the source and destination nodes. In the parallel Ethernet network, there are two disjoint paths available between the source and destination nodes.

However in the fault-tolerant Ethernet network, there are two disjoint paths as well as two additional redundant paths between the source and destination nodes a shown in Figure 37.

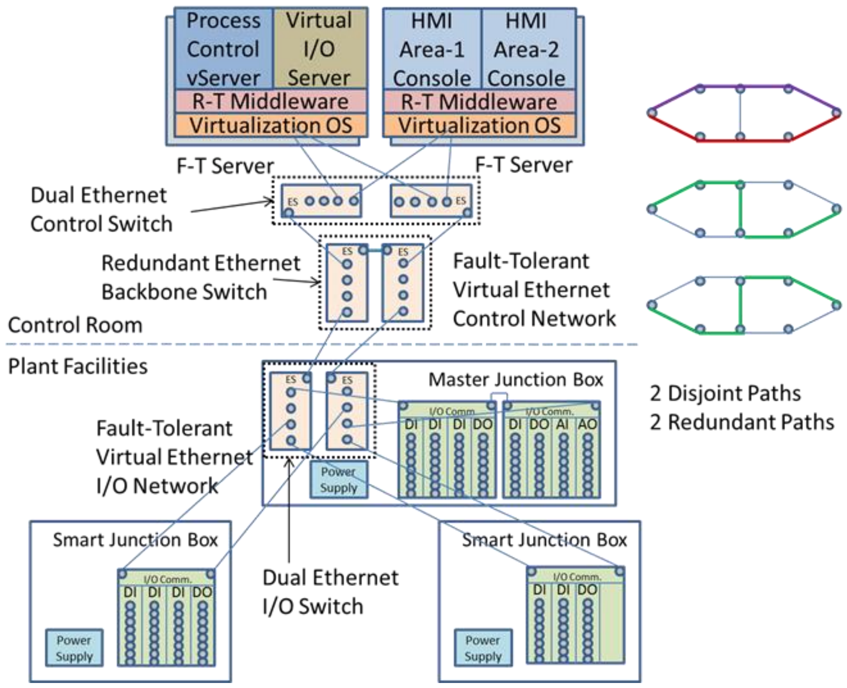


Figure 37 - Fault-Tolerant vCAP with Fault-Tolerant Network Architecture

Therefore, there are four redundant paths between the source and destination nodes. This architecture improves the return on asset by taking advantage of the additional communication hardware used in the parallel Ethernet network described above. The fault-tolerant vCAP with fault-tolerant network represents the final vCAP architecture as shown in Figure 38. The fault-tolerant I/O servers are physically connected to dual Ethernet switches for the virtual local area I/O network; however, the virtual I/O servers are common components in both virtual local area networks.

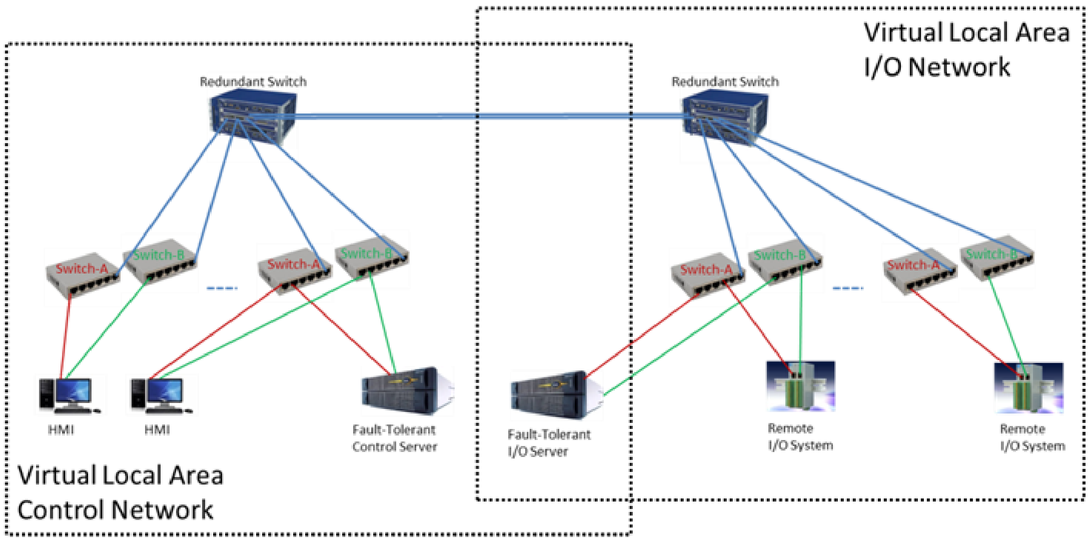


Figure 38 - vCAP Architecture

The fault-tolerant network consists of interconnected redundant Ethernet switches as the backbone network, multiple dual Ethernet switches for the virtual local area control network, and multiple dual Ethernet switches for the virtual local area I/O network as shown in Figure 39. Using a standard 4-port Ethernet switch, the first control level can support up to 8 control segments, the second control level can support up to 32 control segments, and the third control level can support up to 128 control segments. Each control segment can support up to four devices such as control servers, I/O servers, HMI servers, data servers, etc. with a total of 512 devices. Similarly for the I/O network, the first I/O level can support up to 8 I/O segments, the second I/O level can support up to 32 I/O segments, and the third I/O level can support up to 128 I/O segments. Each I/O segment can support up to four I/O systems with a total of 512 I/O systems.

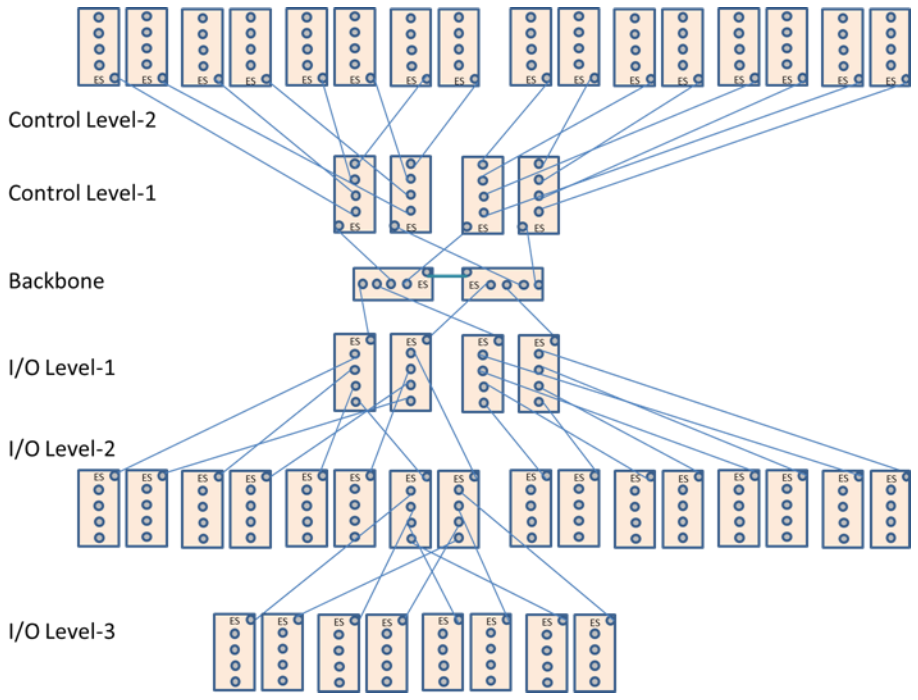


Figure 39 - vCAP Fault-Tolerant Network Structure

The fault-tolerant Ethernet network of the vCAP can be established based on different standard Ethernet switches such as 8-port or 16-port to meet the number of control and I/O segments requirement and minimize the number of levels.

4.4 Reliability Analysis

The terminal reliability of communication network is the probability of having at least one available path between any sources to any destinations. The terminal reliability is calculated using the combinatorial series and parallel models assuming that the failure probabilities of different components of the system are independent. It is also assumed that no communication frames can be routed through a faulty Ethernet switch, and all links connected to a faulty switch are useless. Under the above conditions, the following formulas describing the terminal reliability of the systems are developed. Sometimes a "success" diagram is used to describe the operational models of a system network from a source to a destination. Figure 40(a) shows the success diagram for a first level fault-tolerant Ethernet control and I/O network. If the success diagram becomes too complex to evaluate exactly, upper-limit approximation on the network terminal reliability can be used. An upper bound on terminal reliability is R ≤ (1 - $\prod_{i=1}^{i=RP}(1 - R_{path-i})$) where RP is the number of redundant paths available from a source to a destination and R_{path-i} is the serial reliability of path-i. The upper bound on terminal reliability calculated as if all paths were in parallel. This calculation is an upper bound because the paths are not independent. That is the failure of a single networking element affects more than one path. Therefore, this approximation gets closer to the actual terminal reliability when terminal reliability of a path is small. Placing the paths in parallel yields a reliability block diagram (RBD).

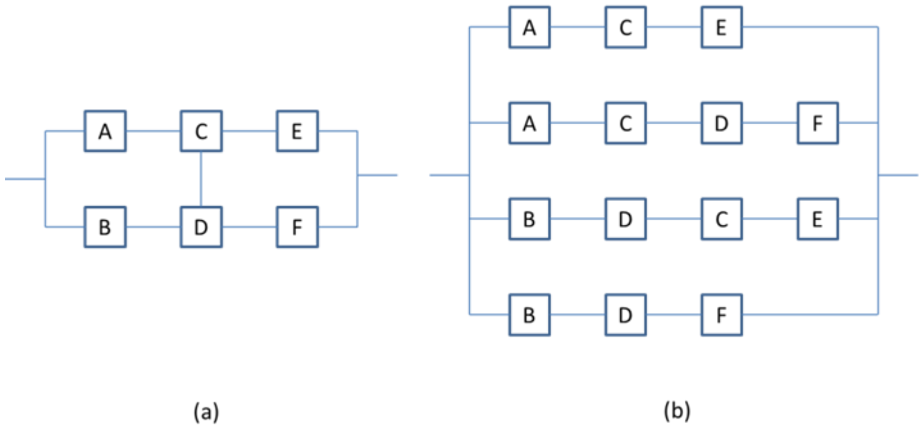
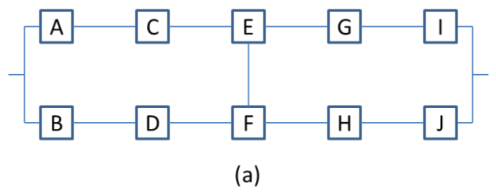


Figure 40 - vCAP Level 1: (a) Success Diagram; (b) Reliability Block Diagram

Figure 40(b) shows the RBD for the success diagram of the first level fault-tolerant Ethernet control and I/O network shown in Figure 40(a). Using the combinatorial series and parallel models, the upper bound on terminal reliability of the first level fault-tolerant Ethernet control and I/O network is calculated as follows R ≤ (1 - $\prod_{i=1}^{i=4}(1 - R_{path-i})$) where $R_{path-1,path-4} = R_{link}^{4} R_{Switch}^{3}$ and $R_{path-2,path-3} = R_{link}^{5} R_{Switch}^{4}$ assuming that the reliability of all communication links are equal and represented by R_{link} and the reliability of all Ethernet switches are equal and represented by R_{Switch}. Therefore, the upper bound on terminal reliability of the first level fault-tolerant Ethernet control and I/O network = $R_{level-1} \leq 1 - (1 - R_{link}^{4} R_{Switch}^{3})^{2}(1 - R_{link}^{5} R_{Switch}^{4})^{2}$. This method would be used to calculate the upper bound terminal reliability of the remaining levels of the fault-tolerant network. Figure 41(b) shows the RBD for the success diagram of the second level fault-tolerant Ethernet control and I/O network shown in Figure 41(a). Using the combinatorial series and parallel models, the upper bound on terminal reliability of the second level fault-tolerant Ethernet control and I/O network is calculated as follows R ≤ $(1 - \prod_{i=1}^{i=4}(1 - R_{path-i}))$ where $R_{path-1,path-4} = R_{link}^{6} R_{Switch}^{5}$ and $R_{path-2,path-3} = R_{link}^{7} R_{Switch}^{6}$ assuming that the reliability of all communication links are equal and represented by R_{link} and the reliability of all Ethernet switches are equal and represented by R_{Switch}. Therefore, the upper bound on terminal reliability of the second level fault-tolerant Ethernet control and I/O network = $R_{level-2} \leq 1 - (1 - R_{link}^{6} R_{Switch}^{5})^{2}(1 - R_{link}^{7} R_{Switch}^{6})^{2}$. The general formula for the upper bound on terminal reliability of all level = $R_{level-n} \leq 1 - (1 - R_{link}^{(2n+2)} R_{Switch}^{(2n+1)})^{2}(1 - R_{link}^{(2n+3)} R_{Switch}^{(2n+2)})^{2}$.



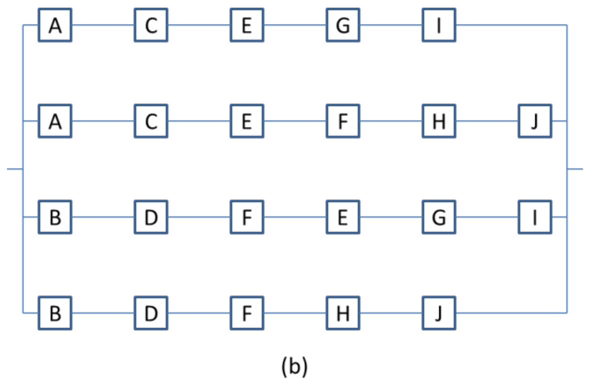


Figure 41 - vCAP Level 2: (a) Success Diagram; (b) Reliability Block Diagram

The overall reliability of vCAP = $1-(1-R_{control\ server})(1-R_{IO\ Server})(1-R_{level-n})(1-R_{Io\ System})$.

Figure 42 shows the reliability of the I/O communication network of the redundant PLC labeled as simplex network and the reliability envelop of the control and I/O communication network of the vCAP with a lower bound labeled as parallel network and an upper bound labeled as fault-tolerant network. It is clear that vCAP is providing higher reliability for the communication network over the conventional redundant PLC.

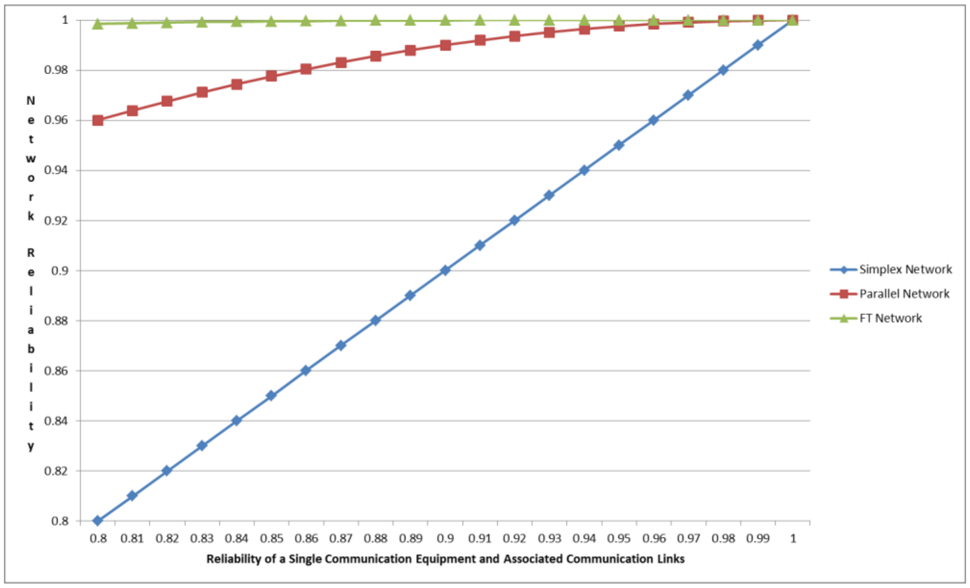


Figure 42 - Control and I/O Network Reliability

4.5 Life Cycle Cost Analysis

The life cycle cost of a control system includes the initial capital cost and the annual capital cost to sustain the system operation throughout the life span of the processing facility. The initial capital cost includes the design phase, detailed engineering phase, implementation phase, installation phase, and commissioning and startup phase. The annual capital cost includes the cost for spare parts, technical support, training, and the implementation of any required hardware and/or software revisions.

The following are the economic model assumptions:

- The life span of the processing facility is 45 years.
- The life span of the proprietary PLC control system is 15 years.

- For the proprietary PLC solution, a total control systems replacement is required every 15 years. Therefore, 2 complete control systems replacements are required within the life span of the processing facility, during the 15th year and during the 30th year.
- The cost escalation factor is 1% every year.
- The total initial capital cost of a control system with 25,000 input/output signals is 25 million US dollars, average of US$ 1000 per I/O signal.
- The annual cost of the contract to manage and maintain the control system for sustaining the processing operation is 0.5% of the initial cost of the control systems, subject to the annual escalation cost factor. Therefore, the contract cost for the first year is US$ 125,000 and for the second year is US$ 126,250 due to the incremental cost escalation.

The total cost of the control systems over the life span of the processing facility based on the proprietary PLC is US$ 94,159,587 and based on the proposed heterogeneous automation controller is US$ 32,060,134. Using the new standard solution to avoid obsolescence challenges results in an approximately 66% cost saving throughout the life span of the processing facility.

4.6 Security Analysis

The proposed controller includes six compulsory security protection mechanisms: (1) Communication between the controller and associated process interface systems are physically isolated from any other communication networks, (2) Controller equipment is located in a physically secured process interface building with key lock, (3) Controller equipment is physically enclosed in secured system cabinets in the secured process interface building with key locks, (4) Controller CPU module has a physical key switch with three positions as (Memory Protect ON, Data Change, and Memory Protect Off), (5) Each process interface system has a physical key switch with three positions as (Memory Protect ON, Data Change, and Memory Protect Off), (6) Communication between the controller and associated process interface systems are secured based on an exclusive publish/subscribe relationship for each data point.

CHAPTER 5

PERFORMANCE ANALYSIS

5.1 Testing Methodology

The model of the proposed vCAP has been evaluated empirically using:

- A small scale prototype of the vCAP to demonstrate the validity of the new virtual control concept and compare its normal steady state operation to that of a conventional proprietary PLC based on a common sequencing control application such as the automation of instrument air dehydrator utilized in oil and gas processing facilities.
- Software based simulation model to demonstrate the performance sustainability of the vCAP while growing in size based on the number of I/O signals and the number of nodes.

For demonstrating the vCAP concept, the instrument air dehydrator control application is configured in both the vCAP model and the conventional PLC. The testing is focused on the proper control sequencing of the process with different time scan resolution ranging from 1000ms down to 100ms with a decrement of 100ms in each subsequent run. The measuring criteria are the proper control sequencing and the number of scan cycle overrun alarms when the actual maximum control loop scan time exceed the predefined time scan resolution. Figure 43 shows the architecture of the vCAP model and the conventional PLC for conducting the empirical tests to demonstrate the concept and to demonstrate the performance sustainability.

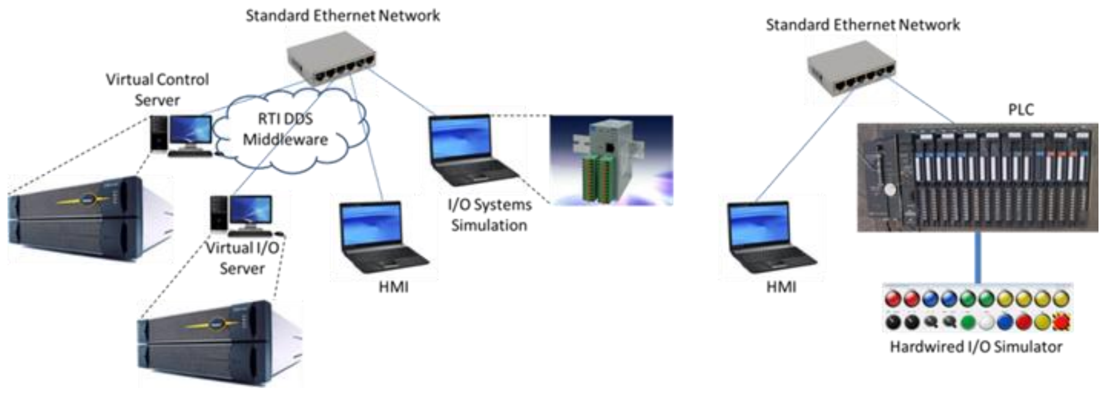
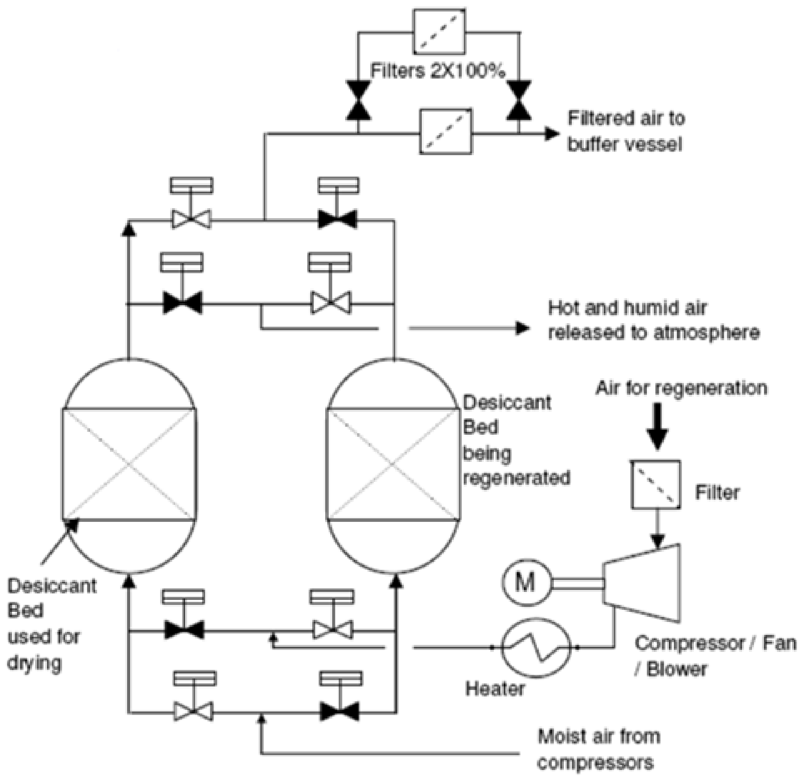


Figure 43 - vCAP Testing Model

5.2 Small Scale vCAP Prototype

An instrument air dehydrator is a device for removing water vapor from compressed air. Compressed air dryers are commonly found in a wide range of industrial and commercial facilities. The process of air compression concentrates atmospheric water vapor. This raises the dew point of the compressed air relative to free atmospheric air and leads to condensation within pipes as the compressed air cools downstream of the compressor. Excessive water in compressed air, in either the liquid or vapor phase, can cause a variety of operational problems including malfunction of process control instruments utilizing compressed air for mechanical movement such as opening or closing pneumatic control valves. There are various types of compressed air dehydrators. Regenerative desiccant dehydrators, shown in Figure 44, are widely used in oil and gas process applications. The compressed air is passed through dual pressure vessels filled with a media such as activated alumina, silica gel, molecular sieve or other desiccant material. This desiccant material attracts the water from the compressed air via adsorption. This is called the drying cycle. During this cycle, the water clings to the desiccant until the desiccant bed becomes saturated. The dehydrator is timed to switch vessels based on a fixed standard NEMA time cycle or dynamically based on dew point threshold measured by moister analyzers. Oftentimes these moister analyzers can save significant amounts of energy which is one of the largest factors when selecting the proper compressed air system.

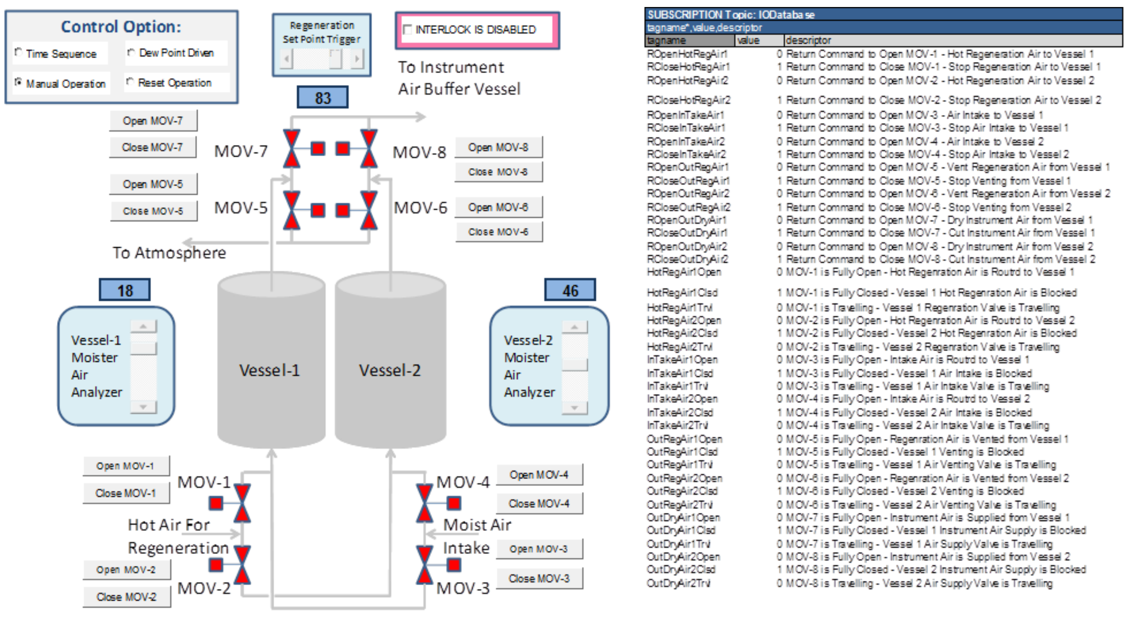


Filters 2X100%
Filtered air to buffer vessel
Hot and humid air released to atmosphere
Air for regeneration
Desiccant Bed being regenerated
Filter
M
Desiccant Bed used for drying
Compressor / Fan / Blower
Heater
Moist air from compressors

Figure 44 - Instrument Air Dehydrator Process Flow Diagram

Once the drying cycle completes some heated compressed air from the system is used to purge the saturated desiccant bed by simply blowing off the water that has adhered to the desiccant. This is called the desiccant regeneration cycle. Once the regeneration cycle completes, the regenerated vessel is put in standby mode.

There are three modes for controlling the instrument air dehydrator process as shown in the top left corner of Figure 45: manual line-up mode, automatic fixed time cycling mode and automatic dynamic time cycling mode based on dew point regeneration threshold.



Control Option:

☐ Time Sequence ☐ Dew Point Driven

[✓ Manual Operation] ☐ Reset Operation

☐ INTERLOCK IS DISABLED

SUBSCRIPTION Topic: IODatabase

tagname*,value,descriptor

tagname	value	descriptor
ROpenHotRegAir1	0	Return Command to Open MOV-1 - Hot Regeneration Air to Vessel 1
RCloseHotRegAir1	1	Return Command to Close MOV-1 - Stop Regeneration Air to Vessel 1
ROpenHotRegAir2	0	Return Command to Open MOV-2 - Hot Regeneration Air to Vessel 2
RCloseHotRegAir2	1	Return Command to Close MOV-2 - Stop Regeneration Air to Vessel 2
ROpenInTakeAir1	0	Return Command to Open MOV-3 - Air Intake to Vessel 1
RCloseInTakeAir1	1	Return Command to Close MOV-3 - Stop Air Intake to Vessel 1
ROpenInTakeAir2	0	Return Command to Open MOV-4 - Air Intake to Vessel 2
RCloseInTakeAir2	1	Return Command to Close MOV-4 - Stop Air Intake to Vessel 2
ROpenOutRegAir1	0	Return Command to Open MOV-5 - Vent Regeneration Air from Vessel 1
RCloseOutRegAir1	1	Return Command to Close MOV-5 - Stop Venting from Vessel 1
ROpenOutRegAir2	0	Return Command to Open MOV-6 - Vent Regeneration Air from Vessel 2
RCloseOutRegAir2	1	Return Command to Close MOV-6 - Stop Venting from Vessel 2
ROpenOutDryAir1	0	Return Command to Open MOV-7 - Dry Instrument Air from Vessel 1
RCloseOutDryAir1	1	Return Command to Close MOV-7 - Cut Instrument Air from Vessel 1
ROpenOutDryAir2	0	Return Command to Open MOV-8 - Dry Instrument Air from Vessel 2
RCloseOutDryAir2	1	Return Command to Close MOV-8 - Cut Instrument Air from Vessel 2
HotRegAir1Open	0	MOV-1 is Fully Open - Hot Regenration Air is Routrd to Vessel 1
HotRegAir1Clsd	1	MOV-1 is Fully Closed - Vessel 1 Hot Regenration Air is Blocked
HotRegAir1Trvl	0	MOV-1 is Travelling - Vessel 1 Regenration Valve is Travelling
HotRegAir2Open	0	MOV-2 is Fully Open - Hot Regenration Air is Routrd to Vessel 2
HotRegAir2Clsd	1	MOV-2 is Fully Closed - Vessel 2 Hot Regenration Air is Blocked
HotRegAir2Trvl	0	MOV-2 is Travelling - Vessel 2 Regenration Valve is Travelling
InTakeAir1Open	0	MOV-3 is Fully Open - Intake Air is Routrd to Vessel 1
InTakeAir1Clsd	1	MOV-3 is Fully Closed - Vessel 1 Air Intake is Blocked
InTakeAir1Trvl	0	MOV-3 is Travelling - Vessel 1 Air Intake Valve is Travelling
InTakeAir2Open	0	MOV-4 is Fully Open - Intake Air is Routrd to Vessel 2
InTakeAir2Clsd	1	MOV-4 is Fully Closed - Vessel 2 Air Intake is Blocked
InTakeAir2Trvl	0	MOV-4 is Travelling - Vessel 2 Air Intake Valve is Travelling
OutRegAir1Open	0	MOV-5 is Fully Open - Regenration Air is Vented from Vessel 1
OutRegAir1Clsd	1	MOV-5 is Fully Closed - Vessel 1 Venting is Blocked
OutRegAir1Trvl	0	MOV-5 is Travelling - Vessel 1 Air Venting Valve is Travelling
OutRegAir2Open	0	MOV-6 is Fully Open - Regenration Air is Vented from Vessel 2
OutRegAir2Clsd	1	MOV-6 is Fully Closed - Vessel 2 Venting is Blocked
OutRegAir2Trvl	0	MOV-6 is Travelling - Vessel 2 Air Venting Valve is Travelling
OutDryAir1Open	0	MOV-7 is Fully Open - Instrument Air is Supplied from Vessel 1
OutDryAir1Clsd	1	MOV-7 is Fully Closed - Vessel 1 Instrument Air Supply is Blocked
OutDryAir1Trvl	0	MOV-7 is Travelling - Vessel 1 Air Supply Valve is Travelling
OutDryAir2Open	0	MOV-8 is Fully Open - Instrument Air is Supplied from Vessel 2
OutDryAir2Clsd	1	MOV-8 is Fully Closed - Vessel 2 Instrument Air Supply is Blocked
OutDryAir2Trvl	0	MOV-8 is Travelling - Vessel 2 Air Supply Valve is Travelling

Figure 45 - Instrument Air Dehydrator HMI Schematic

There are six valid motor operated valve (MOV) line-ups listed in Table 1 along with the command signals to the MOVs controlling the input and output of the pressure vessels: vessel-1 drying, vessel-1 regenerating, vessel-1 standby, vessel-2 drying, vessel-2 regeneration, and vessel-2 standby. Invalid MOV line-ups in manual mode of operation are not permitted unless the interlock override is enabled for troubleshooting purposes.

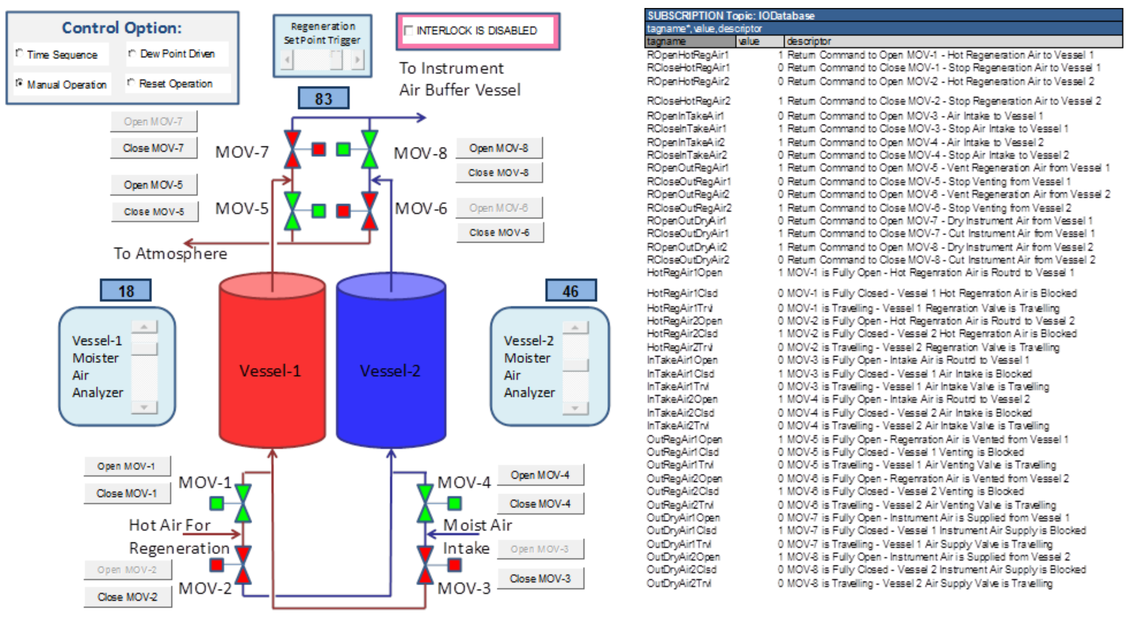
Figure 45 illustrates the third and sixth MOV line-ups corresponding to a complete shutdown of the instrument air dehydrator. On the other hand, Figure 46 illustrates the second and fourth MOV line-ups where vessel-2 is in service and vessel-1 is regenerating. The color of the MOV is red when fully closed and green when fully open.

Table 1 - Instrument Air Dehydrator Valid Line-Ups

#	Line-up Description	MOV-1	MOV-2	MOV-3	MOV-4	MOV-5	MOV-6	MOV-7	MOV-8
1	Vessel-1 is drying	Close	-	Open	Close	Close	-	Open	Close
2	Vessel-1 is regenerating	Open	Close	Close	-	Open	Close	Close	-

3	Vessel-1 standby mode	Close	-	Close	-	Close	-	Close	-
4	Vessel-2 is drying	-	Close	Close	Open	-	Close	Close	Open
5	Vessel-2 is regeneration	Close	Open	-	Close	Close	Open	-	Close
6	Vessel-2 standby mode	-	Close	-	Close	-	Close	-	Close

The MOV body color during the opening and closing operations is yellow while the color of the MOV head during the opening operation is green and red during the closing operation. The color of the vessel is gray while in standby mode, red while regenerating, and blue while in service. The default color of MOV line-up is gray, red when hot air for regeneration is flowing, and blue when moist air intake is flowing.



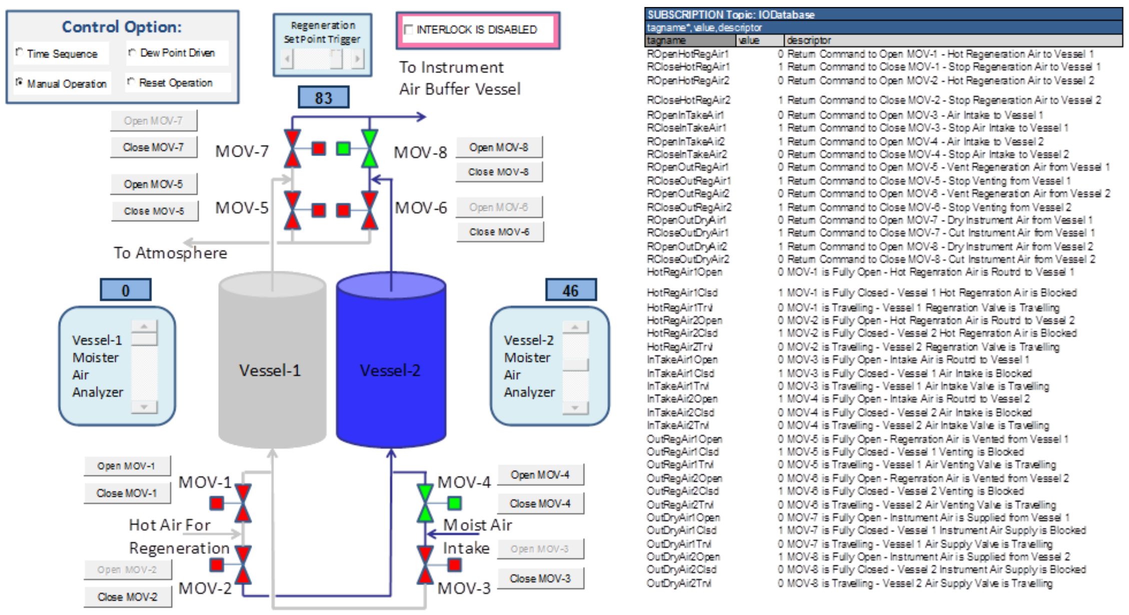
SUBSCRIPTION Topic: IODatabase

tagname*, value, descriptor

tagname	value	descriptor
ROpenHotRegAir1	1	Return Command to Open MOV-1 - Hot Regeneration Air to Vessel 1
RCloseHotRegAir1	0	Return Command to Close MOV-1 - Stop Regeneration Air to Vessel 1
ROpenHotRegAir2	0	Return Command to Open MOV-2 - Hot Regeneration Air to Vessel 2
RCloseHotRegAir2	1	Return Command to Close MOV-2 - Stop Regeneration Air to Vessel 2
ROpenInTakeAir1	0	Return Command to Open MOV-3 - Air Intake to Vessel 1
RCloseInTakeAir1	1	Return Command to Close MOV-3 - Stop Air Intake to Vessel 1
ROpenInTakeAir2	1	Return Command to Open MOV-4 - Air Intake to Vessel 2
RCloseInTakeAir2	0	Return Command to Close MOV-4 - Stop Air Intake to Vessel 2
ROpenOutRegAir1	1	Return Command to Open MOV-5 - Vent Regeneration Air from Vessel 1
RCloseOutRegAir1	0	Return Command to Close MOV-5 - Stop Venting from Vessel 1
ROpenOutRegAir2	0	Return Command to Open MOV-6 - Vent Regeneration Air from Vessel 2
RCloseOutRegAir2	1	Return Command to Close MOV-6 - Stop Venting from Vessel 2
ROpenOutDryAir1	0	Return Command to Open MOV-7 - Dry Instrument Air from Vessel 1
RCloseOutDryAir1	1	Return Command to Close MOV-7 - Cut Instrument Air from Vessel 1
ROpenOutDryAir2	1	Return Command to Open MOV-8 - Dry Instrument Air from Vessel 2
RCloseOutDryAir2	0	Return Command to Close MOV-8 - Cut Instrument Air from Vessel 2
HotRegAir1Open	1	MOV-1 is Fully Open - Hot Regenration Air is Routrd to Vessel 1
HotRegAir1Clsd	0	MOV-1 is Fully Closed - Vessel 1 Hot Regenration Air is Blocked
HotRegAir1Trvl	0	MOV-1 is Travelling - Vessel 1 Regenration Valve is Travelling
HotRegAir2Open	0	MOV-2 is Fully Open - Hot Regenration Air is Routrd to Vessel 2
HotRegAir2Clsd	1	MOV-2 is Fully Closed - Vessel 2 Hot Regenration Air is Blocked
HotRegAir2Trvl	0	MOV-2 is Travelling - Vessel 2 Regenration Valve is Travelling
InTakeAir1Open	0	MOV-3 is Fully Open - Intake Air is Routrd to Vessel 1
InTakeAir1Clsd	1	MOV-3 is Fully Closed - Vessel 1 Air Intake is Blocked
InTakeAir1Trvl	0	MOV-3 is Travelling - Vessel 1 Air Intake Valve is Travelling
InTakeAir2Open	1	MOV-4 is Fully Open - Intake Air is Routrd to Vessel 2
InTakeAir2Clsd	0	MOV-4 is Fully Closed - Vessel 2 Air Intake is Blocked
InTakeAir2Trvl	0	MOV-4 is Travelling - Vessel 2 Air Intake Valve is Travelling
OutRegAir1Open	1	MOV-5 is Fully Open - Regenration Air is Vented from Vessel 1
OutRegAir1Clsd	0	MOV-5 is Fully Closed - Vessel 1 Venting is Blocked
OutRegAir1Trvl	0	MOV-5 is Travelling - Vessel 1 Air Venting Valve is Travelling
OutRegAir2Open	0	MOV-6 is Fully Open - Regenration Air is Vented from Vessel 2
OutRegAir2Clsd	1	MOV-6 is Fully Closed - Vessel 2 Venting is Blocked
OutRegAir2Trvl	0	MOV-6 is Travelling - Vessel 2 Air Venting Valve is Travelling
OutDryAir1Open	0	MOV-7 is Fully Open - Instrument Air is Supplied from Vessel 1
OutDryAir1Clsd	1	MOV-7 is Fully Closed - Vessel 1 Instrument Air Supply is Blocked
OutDryAir1Trvl	0	MOV-7 is Travelling - Vessel 1 Air Supply Valve is Travelling
OutDryAir2Open	1	MOV-8 is Fully Open - Instrument Air is Supplied from Vessel 2
OutDryAir2Clsd	0	MOV-8 is Fully Closed - Vessel 2 Instrument Air Supply is Blocked
OutDryAir2Trvl	0	MOV-8 is Travelling - Vessel 2 Air Supply Valve is Travelling

Figure 46 - Instrument Air Dehydrator: Vessel-1 Regenerating and Vessel-2 Drying

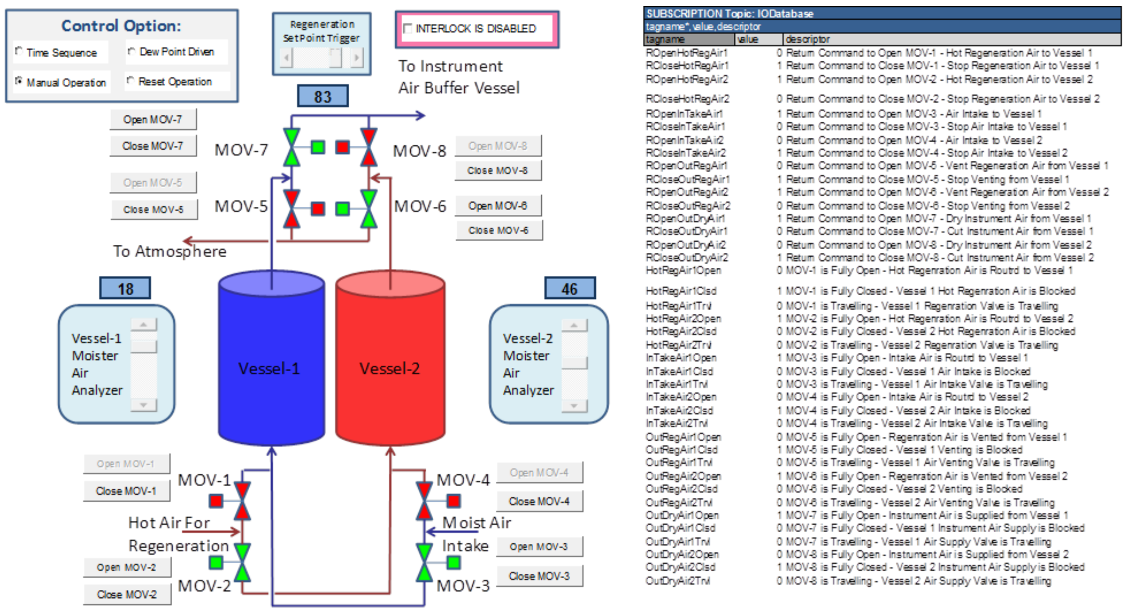
Figure 47 depicts the status once the regeneration cycle of vessel-1 is complete while vessel-2 is still in service. Figure 48 shows the next step when the drying service is switched to vessel-1 and vessel-2 starts regenerating which represents the first and fifth MOV line-ups.



SUBSCRIPTION Topic: IODatabase
tagname*, value, descriptor

tagname	value	descriptor
ROpenHotRegAir1	0	Return Command to Open MOV-1 - Hot Regeneration Air to Vessel 1
RCloseHotRegAir1	1	Return Command to Close MOV-1 - Stop Regeneration Air to Vessel 1
ROpenHotRegAir2	0	Return Command to Open MOV-2 - Hot Regeneration Air to Vessel 2
RCloseHotRegAir2	1	Return Command to Close MOV-2 - Stop Regeneration Air to Vessel 2
ROpenInTakeAir1	0	Return Command to Open MOV-3 - Air Intake to Vessel 1
RCloseInTakeAir1	1	Return Command to Close MOV-3 - Stop Air Intake to Vessel 1
ROpenInTakeAir2	1	Return Command to Open MOV-4 - Air Intake to Vessel 2
RCloseInTakeAir2	0	Return Command to Close MOV-4 - Stop Air Intake to Vessel 2
ROpenOutRegAir1	0	Return Command to Open MOV-5 - Vent Regeneration Air from Vessel 1
RCloseOutRegAir1	1	Return Command to Close MOV-5 - Stop Venting from Vessel 1
ROpenOutRegAir2	0	Return Command to Open MOV-6 - Vent Regeneration Air from Vessel 2
RCloseOutRegAir2	1	Return Command to Close MOV-6 - Stop Venting from Vessel 2
ROpenOutDryAir1	0	Return Command to Open MOV-7 - Dry Instrument Air from Vessel 1
RCloseOutDryAir1	1	Return Command to Close MOV-7 - Cut Instrument Air from Vessel 1
ROpenOutDryAir2	1	Return Command to Open MOV-8 - Dry Instrument Air from Vessel 2
RCloseOutDryAir2	0	Return Command to Close MOV-8 - Cut Instrument Air from Vessel 2
HotRegAir1Open	0	MOV-1 is Fully Open - Hot Regenration Air is Routrd to Vessel 1
HotRegAir1Clsd	1	MOV-1 is Fully Closed - Vessel 1 Hot Regenration Air is Blocked
HotRegAir1Trvl	0	MOV-1 is Travelling - Vessel 1 Regenration Valve is Travelling
HotRegAir2Open	0	MOV-2 is Fully Open - Hot Regenration Air is Routrd to Vessel 2
HotRegAir2Clsd	1	MOV-2 is Fully Closed - Vessel 2 Hot Regenration Air is Blocked
HotRegAir2Trvl	0	MOV-2 is Travelling - Vessel 2 Regenration Valve is Travelling
InTakeAir1Open	0	MOV-3 is Fully Open - Intake Air is Routrd to Vessel 1
InTakeAir1Clsd	1	MOV-3 is Fully Closed - Vessel 1 Air Intake is Blocked
InTakeAir1Trvl	0	MOV-3 is Travelling - Vessel 1 Air Intake Valve is Travelling
InTakeAir2Open	1	MOV-4 is Fully Open - Intake Air is Routrd to Vessel 2
InTakeAir2Clsd	0	MOV-4 is Fully Closed - Vessel 2 Air Intake is Blocked
InTakeAir2Trvl	0	MOV-4 is Travelling - Vessel 2 Air Intake Valve is Travelling
OutRegAir1Open	0	MOV-5 is Fully Open - Regenration Air is Vented from Vessel 1
OutRegAir1Clsd	1	MOV-5 is Fully Closed - Vessel 1 Venting is Blocked
OutRegAir1Trvl	0	MOV-5 is Travelling - Vessel 1 Air Venting Valve is Travelling
OutRegAir2Open	0	MOV-6 is Fully Open - Regenration Air is Vented from Vessel 2
OutRegAir2Clsd	1	MOV-6 is Fully Closed - Vessel 2 Venting is Blocked
OutRegAir2Trvl	0	MOV-6 is Travelling - Vessel 2 Air Venting Valve is Travelling
OutDryAir1Open	0	MOV-7 is Fully Open - Instrument Air is Supplied from Vessel 1
OutDryAir1Clsd	1	MOV-7 is Fully Closed - Vessel 1 Instrument Air Supply is Blocked
OutDryAir1Trvl	0	MOV-7 is Travelling - Vessel 1 Air Supply Valve is Travelling
OutDryAir2Open	1	MOV-8 is Fully Open - Instrument Air is Supplied from Vessel 2
OutDryAir2Clsd	0	MOV-8 is Fully Closed - Vessel 2 Instrument Air Supply is Blocked
OutDryAir2Trvl	0	MOV-8 is Travelling - Vessel 2 Air Supply Valve is Travelling

Figure 47 - Instrument Air Dehydrator: Vesel-1 Standby and Vesel-2 Drying



SUBSCRIPTION Topic: IODatabase
tagname*, value, descriptor

tagname	value	descriptor
ROpenHotRegAir1	0	Return Command to Open MOV-1 - Hot Regeneration Air to Vessel 1
RCloseHotRegAir1	1	Return Command to Close MOV-1 - Stop Regeneration Air to Vessel 1
ROpenHotRegAir2	1	Return Command to Open MOV-2 - Hot Regeneration Air to Vessel 2
RCloseHotRegAir2	0	Return Command to Close MOV-2 - Stop Regeneration Air to Vessel 2
ROpenInTakeAir1	1	Return Command to Open MOV-3 - Air Intake to Vessel 1
RCloseInTakeAir1	0	Return Command to Close MOV-3 - Stop Air Intake to Vessel 1
ROpenInTakeAir2	0	Return Command to Open MOV-4 - Air Intake to Vessel 2
RCloseInTakeAir2	1	Return Command to Close MOV-4 - Stop Air Intake to Vessel 2
ROpenOutRegAir1	0	Return Command to Open MOV-5 - Vent Regeneration Air from Vessel 1
RCloseOutRegAir1	1	Return Command to Close MOV-5 - Stop Venting from Vessel 1
ROpenOutRegAir2	1	Return Command to Open MOV-6 - Vent Regeneration Air from Vessel 2
RCloseOutRegAir2	0	Return Command to Close MOV-6 - Stop Venting from Vessel 2
ROpenOutDryAir1	1	Return Command to Open MOV-7 - Dry Instrument Air from Vessel 1
RCloseOutDryAir1	0	Return Command to Close MOV-7 - Cut Instrument Air from Vessel 1
ROpenOutDryAir2	0	Return Command to Open MOV-8 - Dry Instrument Air from Vessel 2
RCloseOutDryAir2	1	Return Command to Close MOV-8 - Cut Instrument Air from Vessel 2
HotRegAir1Open	0	MOV-1 is Fully Open - Hot Regenration Air is Routrd to Vessel 1
HotRegAir1Clsd	1	MOV-1 is Fully Closed - Vessel 1 Hot Regenration Air is Blocked
HotRegAir1Trvl	0	MOV-1 is Travelling - Vessel 1 Regenration Valve is Travelling
HotRegAir2Open	1	MOV-2 is Fully Open - Hot Regenration Air is Routrd to Vessel 2
HotRegAir2Clsd	0	MOV-2 is Fully Closed - Vessel 2 Hot Regenration Air is Blocked
HotRegAir2Trvl	0	MOV-2 is Travelling - Vessel 2 Regenration Valve is Travelling
InTakeAir1Open	1	MOV-3 is Fully Open - Intake Air is Routrd to Vessel 1
InTakeAir1Clsd	0	MOV-3 is Fully Closed - Vessel 1 Air Intake is Blocked
InTakeAir1Trvl	0	MOV-3 is Travelling - Vessel 1 Air Intake Valve is Travelling
InTakeAir2Open	0	MOV-4 is Fully Open - Intake Air is Routrd to Vessel 2
InTakeAir2Clsd	1	MOV-4 is Fully Closed - Vessel 2 Air Intake is Blocked
InTakeAir2Trvl	0	MOV-4 is Travelling - Vessel 2 Air Intake Valve is Travelling
OutRegAir1Open	0	MOV-5 is Fully Open - Regenration Air is Vented from Vessel 1
OutRegAir1Clsd	1	MOV-5 is Fully Closed - Vessel 1 Venting is Blocked
OutRegAir1Trvl	0	MOV-5 is Travelling - Vessel 1 Air Venting Valve is Travelling
OutRegAir2Open	1	MOV-6 is Fully Open - Regenration Air is Vented from Vessel 2
OutRegAir2Clsd	0	MOV-6 is Fully Closed - Vessel 2 Venting is Blocked
OutRegAir2Trvl	0	MOV-6 is Travelling - Vessel 2 Air Venting Valve is Travelling
OutDryAir1Open	1	MOV-7 is Fully Open - Instrument Air is Supplied from Vessel 1
OutDryAir1Clsd	0	MOV-7 is Fully Closed - Vessel 1 Instrument Air Supply is Blocked
OutDryAir1Trvl	0	MOV-7 is Travelling - Vessel 1 Air Supply Valve is Travelling
OutDryAir2Open	0	MOV-8 is Fully Open - Instrument Air is Supplied from Vessel 2
OutDryAir2Clsd	1	MOV-8 is Fully Closed - Vessel 2 Instrument Air Supply is Blocked
OutDryAir2Trvl	0	MOV-8 is Travelling - Vessel 2 Air Supply Valve is Travelling

Figure 48 - Instrument Air Dehydrator: Vessel-1 Drying and Vessel-2 Regenerating

The conventional proprietary PLC set up includes one CPU controller, one 24 VDC power supply, one 32-channel 24 VDC digital input module, one 32-channel 24 VDC digital output module, one 8-channel 4-20mA analog input module, one 16-port 10/100 Mbps fast Ethernet switch, one hardwired I/O simulator panel with 32 switches, 32 lamp indicators and 8 analog input potentiometers, and one Lenovo Thinkpad, i7-4600U, 2.1GHz serving as Engineering workstation. The PLC program includes two sections, sequence control and MOVs' operation simulation. The sequence control includes two modes of operation, fixed and dynamic time cycle control. For simulation purposes, the fixed time is 60 seconds for drying service and 30 seconds for regeneration process. The dynamic time cycle control is based on moister analyzer threshold set points, 83% for the near saturation state and 2% for the completion of desiccant regeneration process. The MOVs' operation simulation includes four states per MOV, fully open, fully closed, opening, and closing. The linear traveling time of each MOV from fully closed to fully open and vice versa is 10 seconds. The simulation panel is hardwired to 20 digital input signals representing the open/close pushbutton commands per MOV, manual/automatic control selector switch, fixed/dynamic cycling selector switch, enable/disable interlock override switch, and process shutdown pushbutton. The simulation panel is also hardwired to 29 digital output signals representing one common alarm for bad MOV line-ups, two state light indicators per vessel, red and blue, and three state light indicators per MOV, green, yellow and red. The MOV green light represents the fully open state. The MOV red light represents the fully closed state. The MOV green and yellow lights represent the opening state. The MOV red and yellow lights represent the closing state. The vessel red light represents the regeneration process state. The vessel blue light represents the drying process state. Vessel standby state is represented by having both red and blue lights off. The simulation panel is also hardwired to 4 analog input signals representing one moisture analyzer per vessel, one potentiometer for setting a common regeneration trigger set point, and one potentiometer for setting a common regeneration completion set point. The communication between the PLC and the Engineering workstation is proprietary.

The vCAP set up, shown in Figure 49, includes one Lenovo Thinkpad, i7-4600U, 2.1GHz serving as the HMI workstation with Windows 7 operating system and Microsoft Excel application for the operator interface, one Lenovo Thinkpad, i7-3667, 2GHz serving as the controller with Windows 7 operating system, one Lenovo Thinkpad, i7-3667, 2GHz serving as the I/O system with Windows 7 operating system, one Lenovo Thinkpad, i7-3667, 2GHz serving as the process simulation with Windows 7 operating system, and one 16-port 10/100 Mbps fast Ethernet switch.

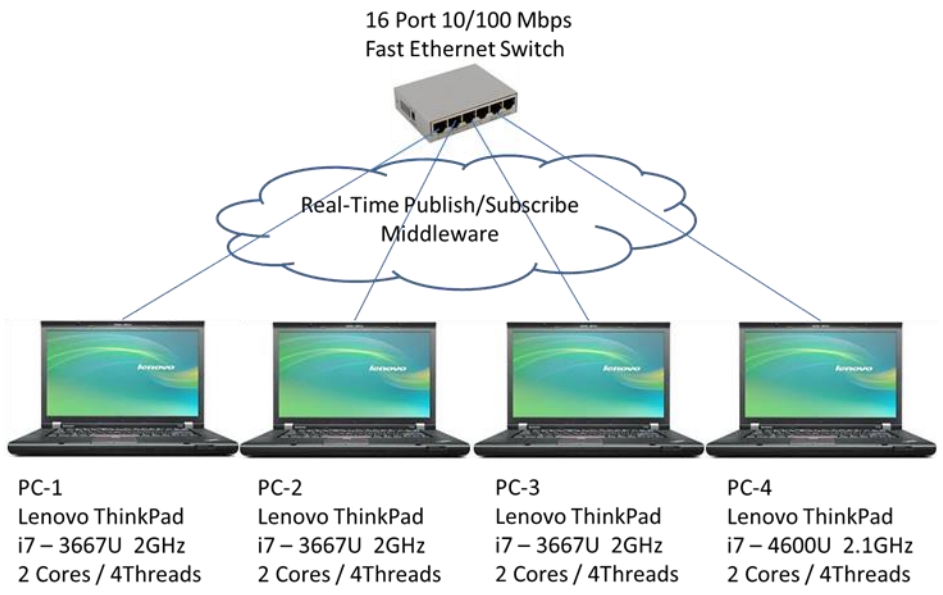


Figure 49 - Virtually Clustered Automation Platform

Real-time Connext DDS professional middleware version 5.1.0.14-i86Win32VS2013 from Real-Time Innovations, Inc. is installed in all Lenovo Thinkpad laptops. VMware Workstation 11.1.0 virtualization for Windows is installed in all Lenovo Thinkpad laptops. The four laptops are connected to one 16-port 10/100 Mbps fast Ethernet switch. The communications among the laptops for the instrument air dehydrator application is based on the real-time publish/subscribe DDS middleware as shown in Figure 50.

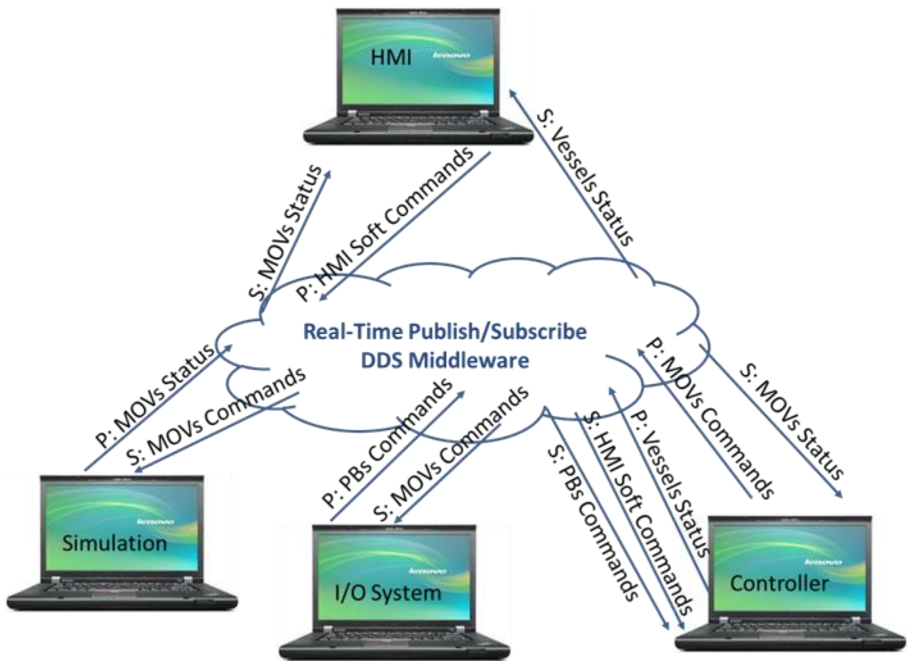


Figure 50 - vCAP Instrument Air Dehydrator Application

The HMI workstation includes digital input subscribers for the status of all MOVs and vessels. It also includes digital output publishers for passing the software-based open/close commands for all MOVs and the operation control modes. Furthermore, it includes analog output publishers for passing the data of the software-based moister analyzers and the required set points for triggering the start and end of the desiccant regeneration cycle. The logic for coloring the pipeline segments is programmed based on received status of all MOVs and vessels and using visual basic macros as part of the Microsoft Excel application.

The controller server includes digital input subscribers for the status of all MOVs and operation mode selector switches as well as the local hardwired and HMI-based pushbutton commands for all MOVs. It also includes analog input subscribers for the emulated moister analyzers and the required set points for triggering the start and end of the desiccant regeneration cycle. The result control actions of the control logic sequence and the associated status of the controlled objects are communicated through digital output publishers for passing the vessels' states and the operation command signals for all MOVs. The sequence control includes two modes of operation, fixed and dynamic time cycle control. For simulation purposes, the fixed time is 60 seconds for drying service and 30 seconds for regeneration process. The dynamic time cycle control is based on moister analyzer threshold set points, 83% for the near saturation state and 2% for the completion of desiccant regeneration process.

The input/output system includes digital input subscribers for the MOVs operation commands from the controller. These signals are transmitted to master control circuits through actual digital output modules for operating the actual MOVs. The I/O system also includes digital output publishers for passing the local hardwired pushbutton command signals of all MOVs.

The process simulation server includes digital input subscribers for the MOVs operation commands from the controller. It also includes digital output publishers for passing the current status of all MOVs. The MOVs' operation simulation includes four states per MOV, fully open, fully closed, opening, and closing. The linear traveling time of each MOV from fully closed to fully open and vice versa is 10 seconds.

The communication relationships among publishers and subscribers are summarized in Table 2.

DDS QoS policies for real-time systems can be used to control and optimize network as well as computing resource to ensure the right information is delivered to the right subscriber at the right time. The default values are used with the following exceptions.

Table 2 - Publish/Subscribe Relationship of vCAP Topics

Publishers	Topics	Subscribers			
		HMI	Simulation	I/O System	Controller
HMI	MOVs Soft Pushbuttons				X
	Operation Control Modes				X
	Moister Analyzer Set Points				X
Simulation	MOVs Status	X			X
I/O System	Local MOVs Pushbuttons				X
Controller	MOVs Operation Commands		X	X	
	Vessels Status	X			

The reliability QoS policy indicates the level of guarantee offered by the DDS in delivering data to subscribers. Possible variants are reliable and best effort. With the selection of reliable parameter in steady-state, the middleware guarantees that all samples in the publisher history will eventually be delivered to all the subscribers. However, the best effort parameter indicates that it is acceptable to not retry propagation of any samples. The reliable option is selected.

The durability QoS policy controls the data availability with respect to late joining publishers and subscribers; specifically the DDS provides the following variants: volatile, transient local, transient, and persistent. With volatile option, there is no need to keep data instances for late joining subscriber. With transient local option, the data instance availability for late joining subscriber is tied to the publisher availability. With transient option, the data instance availability outlives the publisher. With the persistent option, the data instance availability outlives the system restarts. The durability service QoS policy is used to configure the history QoS policy and the resource limits QoS policy used by the fictitious subscriber and publisher used by the “persistence service.” The “persistence service” is the one responsible for implementing the durability QoS policy options of transient and persistence. The persistent option is selected.

The history QoS policy controls whether the DDS should deliver only the most recent value, attempt to deliver all intermediate values, or do something in between. The policy can be configured to provide the following semantics for how many data samples it should keep: keep last and keep all. With keep last option, the DDS will only attempt to keep the most recent “depth” samples of each instance of data identified by its key. However, with the keep all option, the DDS will attempt to keep all the samples of each instance of data identified by its key. The keep all option is selected.

The ownership QoS policy specifies whether it is allowed for multiple publishers to write the same instance of the data and if so, how these modifications should be arbitrated. Possible options are: shared and exclusive. With shared option, multiple publishers are allowed to update the same instance and all the updates are made available to the subscriber. However, the exclusive option indicates that each instance can only be owned by one publisher, but the owner of an instance can change dynamically due to liveliness changes and the selection of the owner is controlled by setting of the ownership strength QoS policy. The ownership strength QoS policy specifies the value of the “strength” used to arbitrate among publishers that attempt to modify the same data instance. The policy applies only if the ownership QoS policy is set to exclusive. The exclusive option is selected.

Thirty empirical tests were conducted for both the conventional PLC and vCAP systems to verify the normal operation for controlling the instrument air dehydrator application and compare the recorded sequence of events and their time stamps. The test changing variable is the scan time resolution starting from 1000ms down to 100ms with a decrement of 100ms in each subsequent test. For each scan time resolution, three tests were conducted to verify the manual operation, fixed time cycling, and dynamic time cycling based on dew point. The performance of the normal operation was verified and the sequence of events was very comparable. The running time for the last test of having 100ms scan time resolution and automatic fixed time cycle control mode was extended for a month of continuous operation and the result was identical to the start of the steady state operation.

5.3 Software-Based vCAP Simulation

The same vCAP model is used for demonstrating the performance sustainability of the vCAP while growing in size based on the number of I/O signals and the number of nodes. The focus of this test is the amount of interaction traffic cross the middleware generated internally from each node as well as the increase in the number of nodes by adding multiple virtual controllers and/or multiple virtual I/O systems. The measuring performance criteria are the average latency and throughput.

5.3.1 Data Simulation and Performance Test Measurement Methodologies

The communication test between a publisher and a subscriber is as follows. The publishing side of the test writes data, total of 30 million data samples, to the middleware as fast as it can. Every time, after writing 1000 data samples to the middleware, it sends a special sample requesting an echo from the subscribing side. On one hand, the publishing application publishes throughput data and at the same time it also subscribes to the latency echoes. On the other hand, the subscribing applications subscribe to the throughput data, in which the echo requests are embedded; they also publish the latency echoes.

5.3.1.1 Communication Latency Measurement for a Single Subscriber

The publisher uses the request for an echo exchange to measure the round-trip latency. The time stamp is logged by the publisher from the start of sending the data sample request until it receives the echo of the data sample back from the subscriber. The communication latency between a publisher and a subscriber is one half of the round-trip latency. The average communication latency between a publisher and a subscriber is the average of the 30 thousand times of latency measurement during one test. The reason for measuring the round-trip latency rather than one-way latency is to overcome the challenge of ensuring accurate clock time synchronization between the publisher and the subscriber.

5.3.1.2 Communication Latency Measurement for Multiple Subscribers

The publisher uses the request for an echo exchange to measure the round-trip latency. There are two methods for measuring the communication latency when there is one publisher and many subscribers; unicast and multicast scenarios.

Unicast Scenario: The time stamp is logged by the publisher from the start of sending the data sample request consecutively to all subscribers until it receives the echo of the data sample back from the last subscriber as illustrated in Figure 51 for two subscribers. The communication latency between a publisher and the last subscriber, second subscriber in this case, is estimated by subtracting the one-way communication latency, as determined in a single publisher and a single subscriber case, from the roundtrip time to the last subscriber. In other words, the one-way latency is equal to T1 – T0 – one-way latency in one-to-one unicast case. The average communication latency between a publisher and the last subscriber is the average of the 30 thousand times of latency measurement during one test.

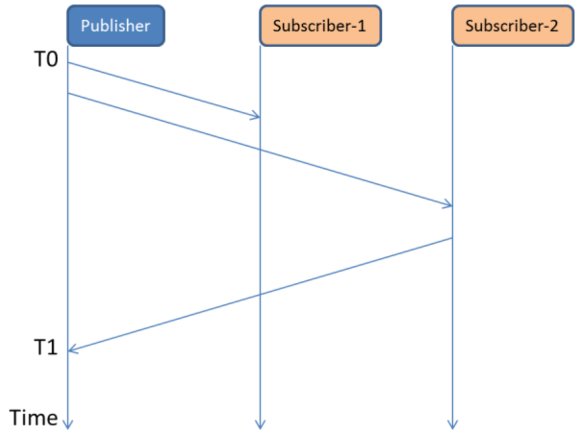


Figure 51 - One-to-Two Unicast Publish-Subscribe Communication Latency Measurement

Multicast Scenario: The time stamp is logged by the publisher from the start of sending the data sample request to all subscribers until it receives the echo of the data sample back from the last subscriber as illustrated in Figure 52 for two subscribers. The communication latency between a publisher and the last subscriber, second subscriber in this case, is

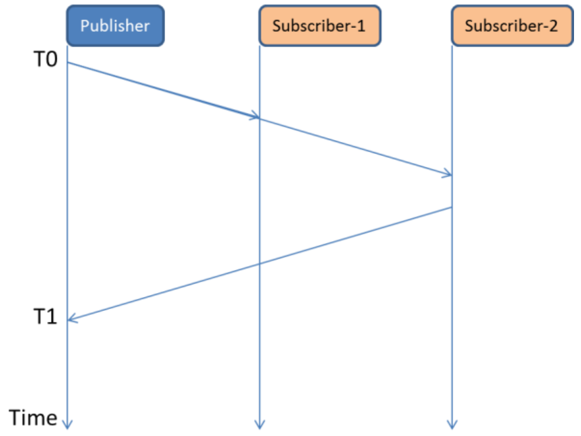


Figure 52 - One-to-Two Multicast Publish-Subscribe Communication Latency Measurement

estimated by subtracting the one-way communication latency, as determined in a single publisher and a single subscriber case, from the roundtrip time to the last subscriber. In other words, the one-way latency is equal to T1 – T0 – one-way latency in one-to-one multicast case. The average communication latency between a publisher and the last subscriber is the average of the 30 thousand times of latency measurement during one test.

5.3.2 Baseline Performance Test Analysis (One Publisher to One Subscriber Scenarios)

Each test scenario is repeated eight times with different data packet size of 100, 200 400,800, 1600, 3200, 6400 and 12800 bytes. The change in data size represents the change in the number of I/O signals. The change in number of publishers and subscribers represent the change in number of controllers and I/O systems. The subscriber measures the throughput by counting the number of received data packets per second and the throughput rate in Megabits per second. Figure 53 depicts the complete vCAP architecture utilized in the performance testing where each of the four laptops is configured with one virtual Ethernet switch and three virtual machines. Total of sixteen machines are interconnected using 100Mbps Ethernet switch and four virtual Ethernet switches. All sixteen machines are configured with RTI real-time Connext DDS middleware. The normal minimum PLC scan time resolution is 100ms. This scan time includes 20ms dedicated for scanning the input signals, 50ms dedicated for solving the control logic, 15ms dedicated for updating the output signals, and 15ms dedicated for diagnostics and housekeeping.

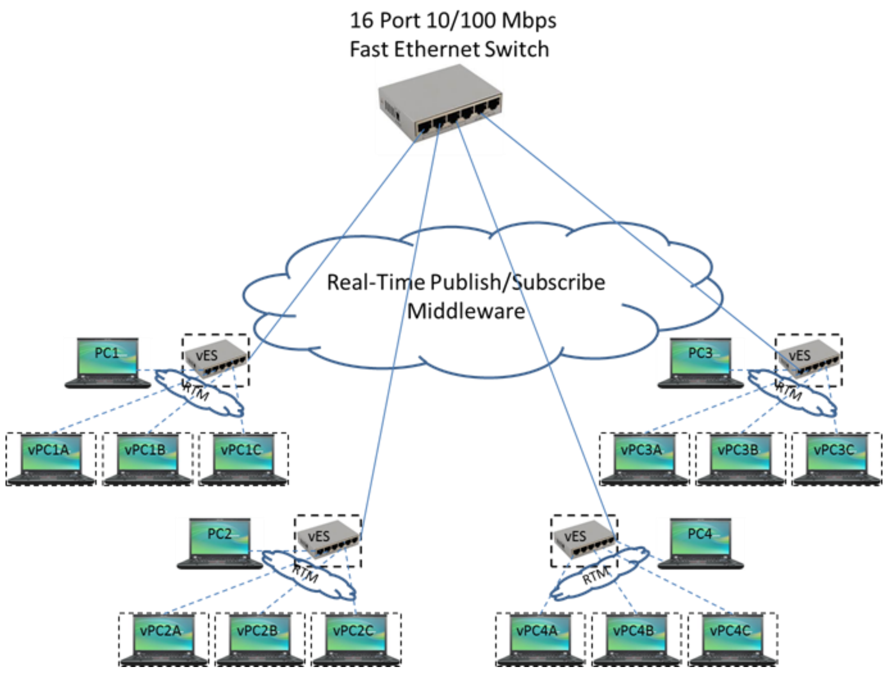


Figure 53 - vCAP Performance Testing Architecture

In other words, a total of 35ms is dedicated for I/O communication services. Therefore, the average communication latency between the controller and the I/O system through the real-time publish/subscribe DDS middleware shall be within 35ms. The baseline performance test is to measure the latency and throughput of one controller and one I/O system within each laptop.

5.3.2.1 Communication Latency and Jitter Analysis within One Laptop

The measured average latency for the three identical laptops as well as the forth laptop is shown in Figure 54. The performance result of the average communication latency between the controller and the I/O system in all laptops is within 1ms, very well below the required scan time resolution while varying the controller size from 100 bytes equivalent to a PLC with 400 digital I/O and 50 analog I/O, to a controller size of 12,800 bytes equivalent to a PLC with 80,000 digital I/O and 2,800 analog I/O. The data shows that communication latency remains consistently low as message size increases.

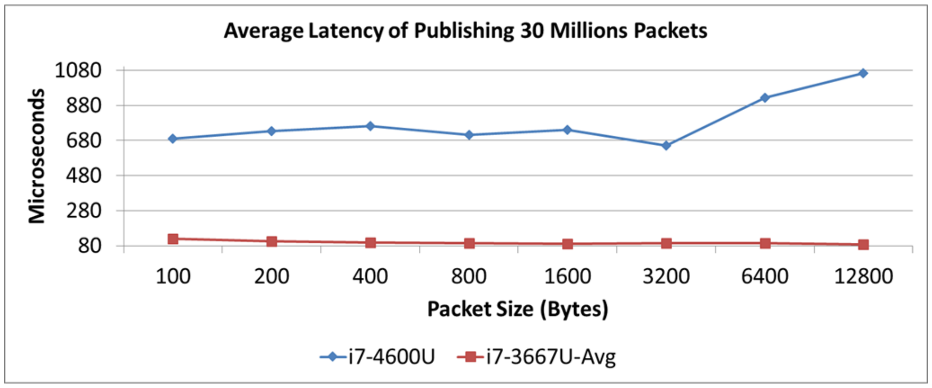


Figure 54 - Average Latency of 1 Controller and 1 I/O System within each Laptop

This is an excellent result showing that the real-time publish/subscribe DDS middleware was able to cope with the huge increase in data loading without any significant impact on the controller performance. Figure 55 shows the latency with jitter analysis for the first laptop. Jitter is the variation in latency as measured in the variability over time of the packet latency across the communication medium. With constant latency, there is no variation or jitter. A system is more deterministic if it exhibits lower jitter. The blue series show the minimum measured latency and green series show the 99th percentile latency. Latency at 99th percentile means that only 1% of the data samples exhibited latency larger than this value. Even at large packet sizes, the variation between the minimum and 99% latency remains consistently low. This shows that the real-time publish/subscribe DDS middleware between the controller and the I/O system exhibits very low jitter and very high determinism, making it suitable for real-time and mission-critical applications.

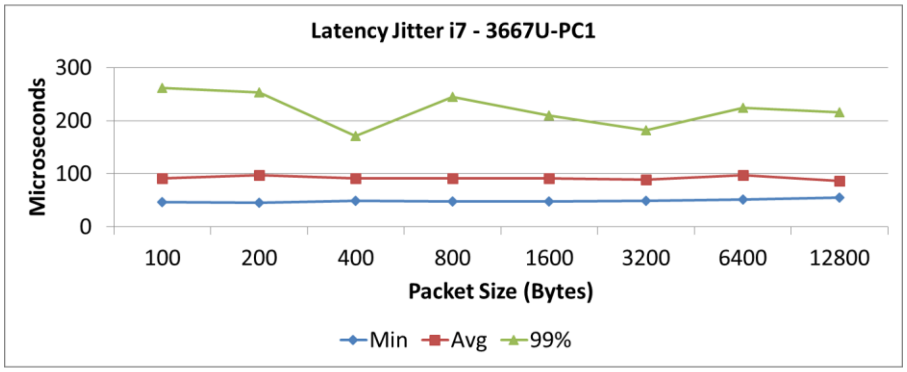


Figure 55 - Average Latency Jitter of 1 Controller and 1 I/O System within PC1

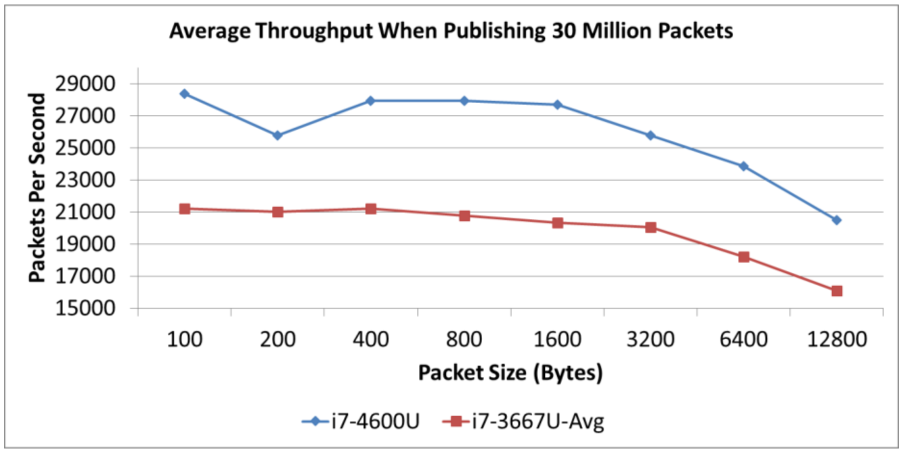
5.3.2.2 Communication Throughput Analysis within One Laptop

For the throughput analysis, the publisher sends data to one subscriber application. The performance test goes through the following phases:

- The publisher signals the subscriber application that it will commence, and then starts its own clock. The duration of the test is based on the number of data samples to be written to the middleware; in this case it is 30 million packets.
- The subscriber starts measuring the number of data samples received.
- After the desired duration is over, the publisher signals the subscriber that the experiment is over. The subscriber will then divide the number of samples received by the elapsed time to report the throughput observed at the receiver.

Maximum throughput is achieved when the publisher sends as fast as the subscriber can handle messages without dropping a packet. That is, the maximum throughput is obtained somewhere between the publisher sending too slowly, not maximizing the available pipe, and the publisher swamping the subscriber, overflowing the pipe. For this reason, the test makes the publisher try a range of sending rates. For the absolute maximum throughput to be observed, the optimal sending rate must be in the range. The measured average throughput bandwidth between one controller and one I/O system for the three identical laptops as well as the forth laptop measured in packets per second and Megabits per second is shown in Figure 56. The graph shows sustainable publish/subscribe throughput bandwidth between one controller and one I/O system within each laptop in terms of packets per second and Megabits per second.

Obviously, the slight decrease in the throughput in terms of number of number of packets per is due to the increase in transmission time of each packet. However, the throughput bandwidth in terms of Megabits per second increases significantly with the increase size of the packet. This indicates that the real-time DDS middleware was able to cope with the huge increase in data loading and fully utilized available data bus communication bandwidth between the controller and the I/O system. In other words, it does not impose any inherent limit on the aggregate data messaging capacity, making it suitable for scalable automation platforms.



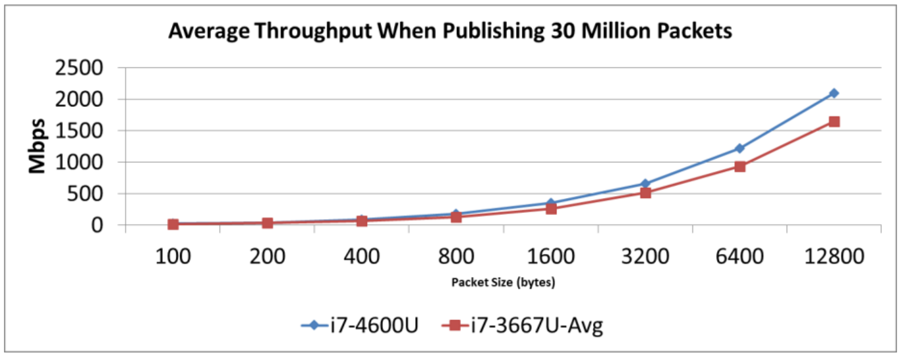
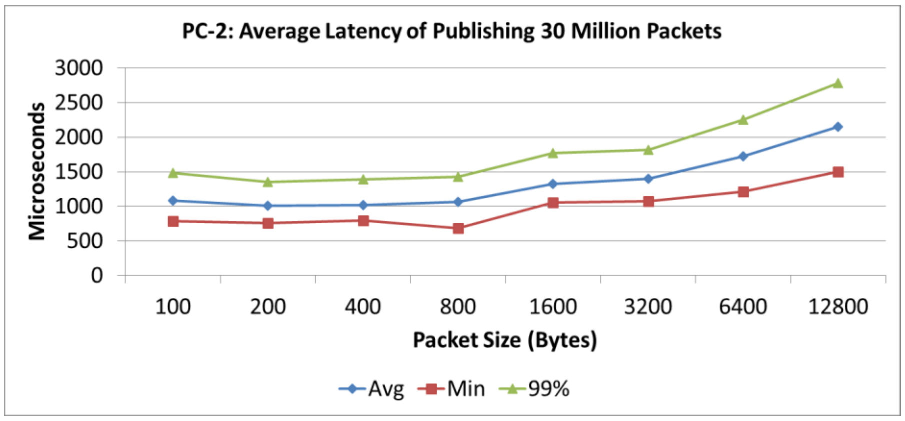


Figure 56 - Average Throughput of 1 Controller and 1 I/O System within each Laptop

5.3.3 Impact Analysis of using I/O System and Virtual Machine for Controller

The set up for the next performance test configuration is to move the controller application to a virtual machine within the same laptop. The communication between the virtual controller and the I/O system is implemented through a virtual Ethernet switch using real-time publish/subscribe DDS middleware. The measured average latency for the two identical laptops is shown in Figure 57.



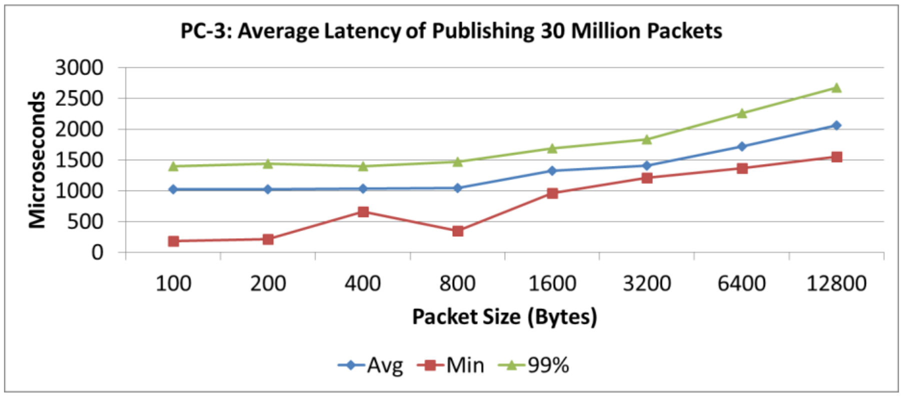


Figure 57 - Average Latency of 1 Virtual Controller and 1 I/O System within each Laptop

The performance result of the average communication latency between the virtual controller and the I/O system in all laptops is within 2ms, very well below the required scan time resolution while varying the controller size from 100 bytes to 12,800 bytes. Again, the data shows that communication latency remains consistently low as message size increases in both cases and the variation between the minimum and 99% latency remains consistently low resulting in very low jitter and very high determinism. Although the virtualization of the controller increased the latency by 100% to 2ms due to the middleware communication through the virtual Ethernet switch, it certainly improved the maintainability of the controller since the operation of the virtual machine is independent of any operations with the host machine. It also improves the economy of scale and utilization of the controller since the memory size and allocated CPU cores of the virtual machine are not fixed and can be adjusted easily to meet the application demand.

The measured average throughput bandwidth between one virtual controller and one I/O system for the two identical laptops measured in packets per second and Megabits per second is shown in Figure 58. The graph compares the result of this test and the baseline test result observed in the previous test. It can be observed that when the packet size is less than 1,600 bytes, the measured throughput in packets per second is slightly higher than the rate resulted when the real-time middleware communication between the controller and I/O system is transported through shared memory. As the packet size increases above 1,600 bytes, the throughput performance in terms of packets per second decreases slightly even below the baseline result due to the increase of the virtual Ethernet communication overhead and transmission time.

This observation is identical for both laptops carried the same test independently. Similarly, the increase in throughput bandwidth in Megabits per second is slightly lower than the baseline when the packet size increases above 1,600 bytes. Again, this indicates that the real-time DDS middleware was able to cope with the huge increase in data loading and fully utilized available virtual communication bandwidth between the virtual controller and the I/O system.

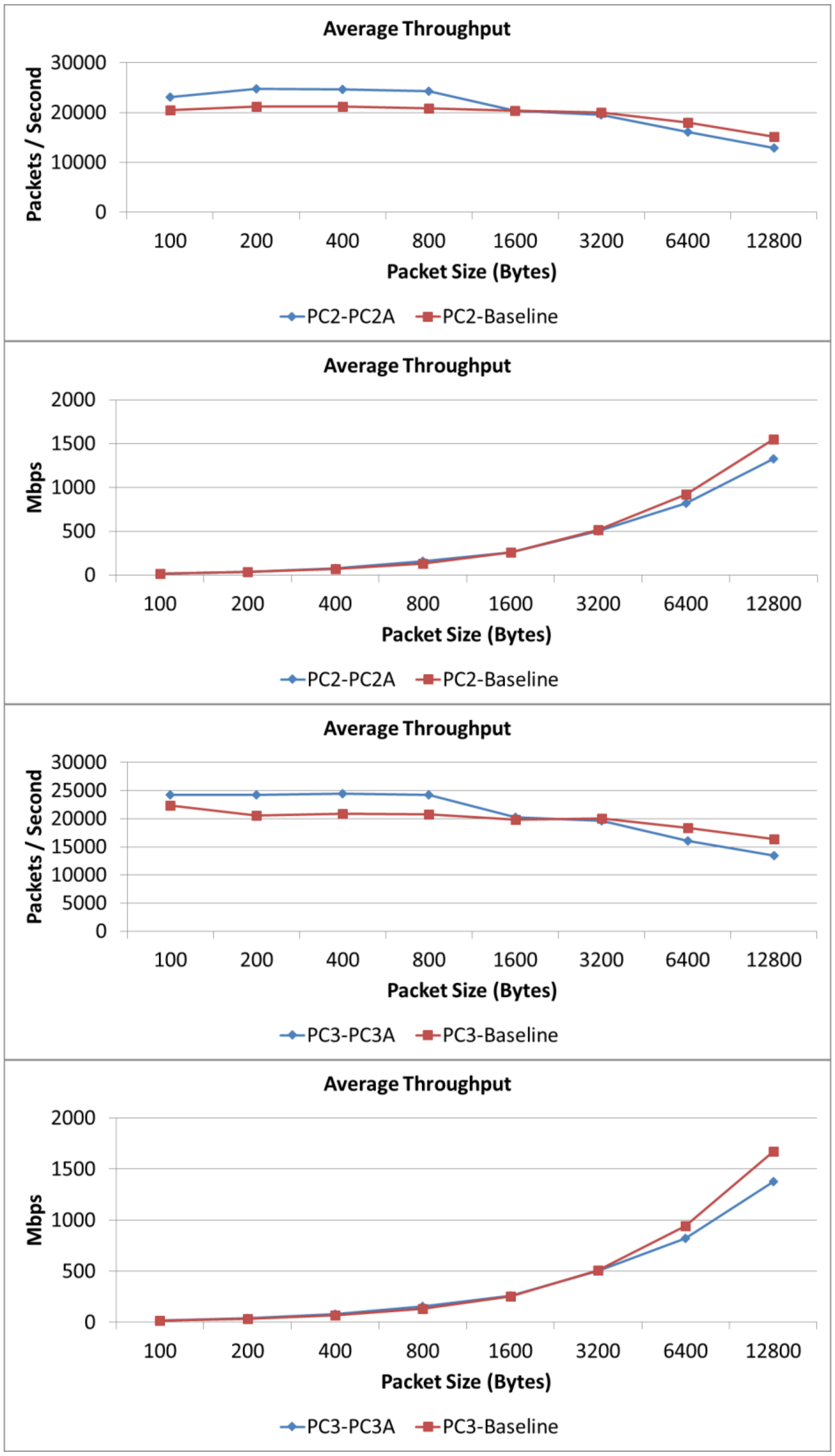


Figure 58 - Average Throughput of 1 Virtual I/O System and 1 Controller within each Laptop

5.3.4 Impact Analysis of using Virtual I/O System and Virtual Controller

The set up for the next performance test configuration is to move the I/O system to a virtual machine within the same laptop. The overall test set up includes two virtual machines connected to one virtual Ethernet switch in the same laptop. The communication between the virtual controller and the virtual I/O system is implemented through a virtual Ethernet switch using real-time publish/subscribe DDS middleware. The measured average latency and throughput is shown in Figure 59. The performance result of the average communication latency between the virtual controller and the virtual I/O system is within 1ms, very well below the required scan time resolution while varying the controller size from 100 bytes to 12,800 bytes. Again, the data shows that communication latency remains consistently low as message size increases in both cases and the variation between the minimum and 99% latency remains consistently low resulting in very low jitter and very high determinism. As the packet size increases above 800 bytes, the throughput performance in terms of packets per second decreases slightly even below the baseline result due to the increase of the virtual Ethernet communication overhead and transmission time. Similarly, the increase in throughput bandwidth in Megabits per second is slightly lower than the baseline when the packet size increases above 800 bytes. The real-time DDS middleware was able to fully utilize available virtual communication bandwidth between the virtual controller and the virtual I/O system. Figure 60 compares the performance of the baseline controller, virtual controller and virtual system. The baseline controller includes one controller and one I/O system in the same laptop. The virtual controller includes one I/O system and one virtual machine hosting the controller application and connected through a virtual Ethernet switch.

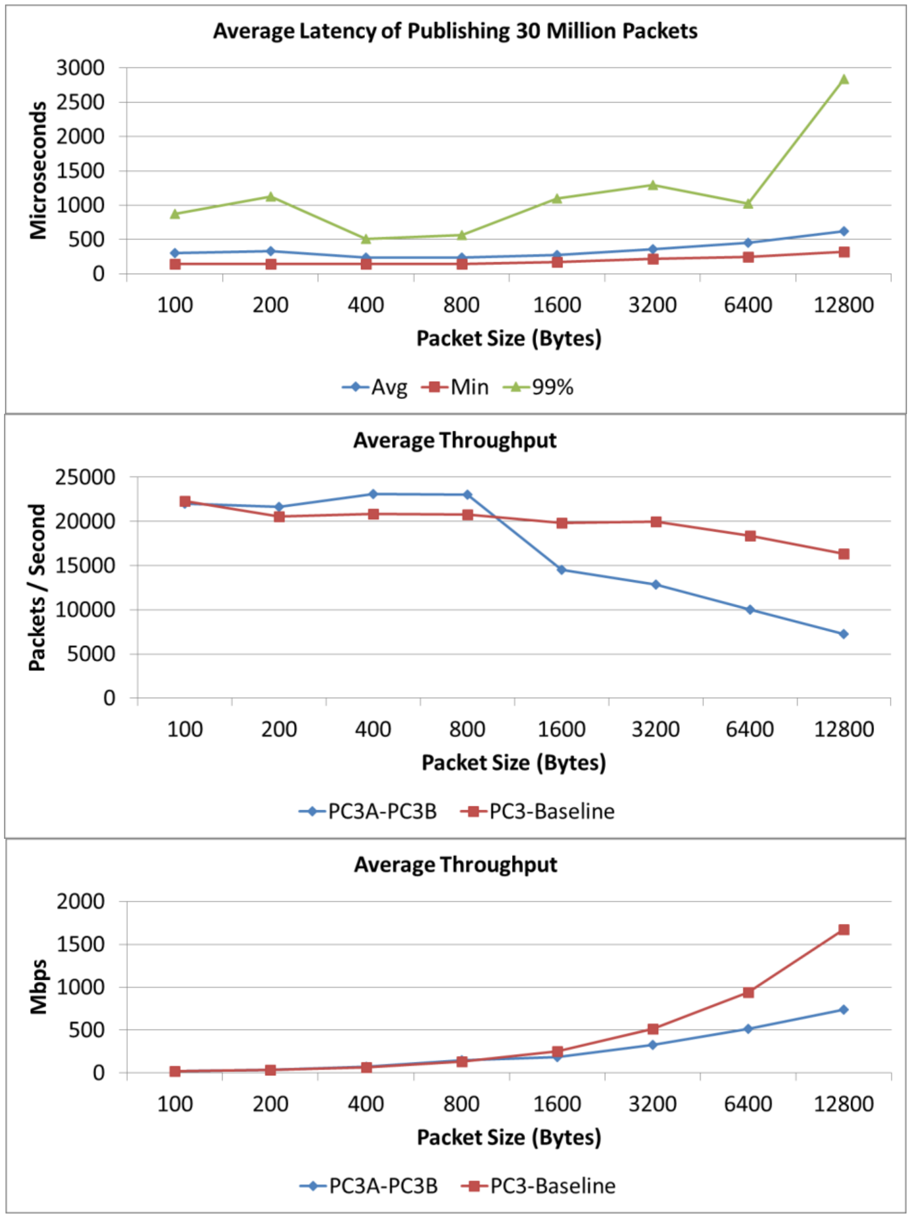


Figure 59 - Average Latency and Throughput of 1 Virtual I/O System and 1 Virtual Controller

The virtual system includes one virtual machine hosting the controller application and one virtual machine hosting the I/O system; both connected through a virtual Ethernet switch. From communication latency prospective, the baseline and the virtual system are very close and within 1ms; however, the virtual controller is within 2ms. On the other hand, from throughput perspective with large packet size data higher than 1,600 bytes, the baseline is slightly higher than the virtual controller; but approximately 100% higher than the virtual system.

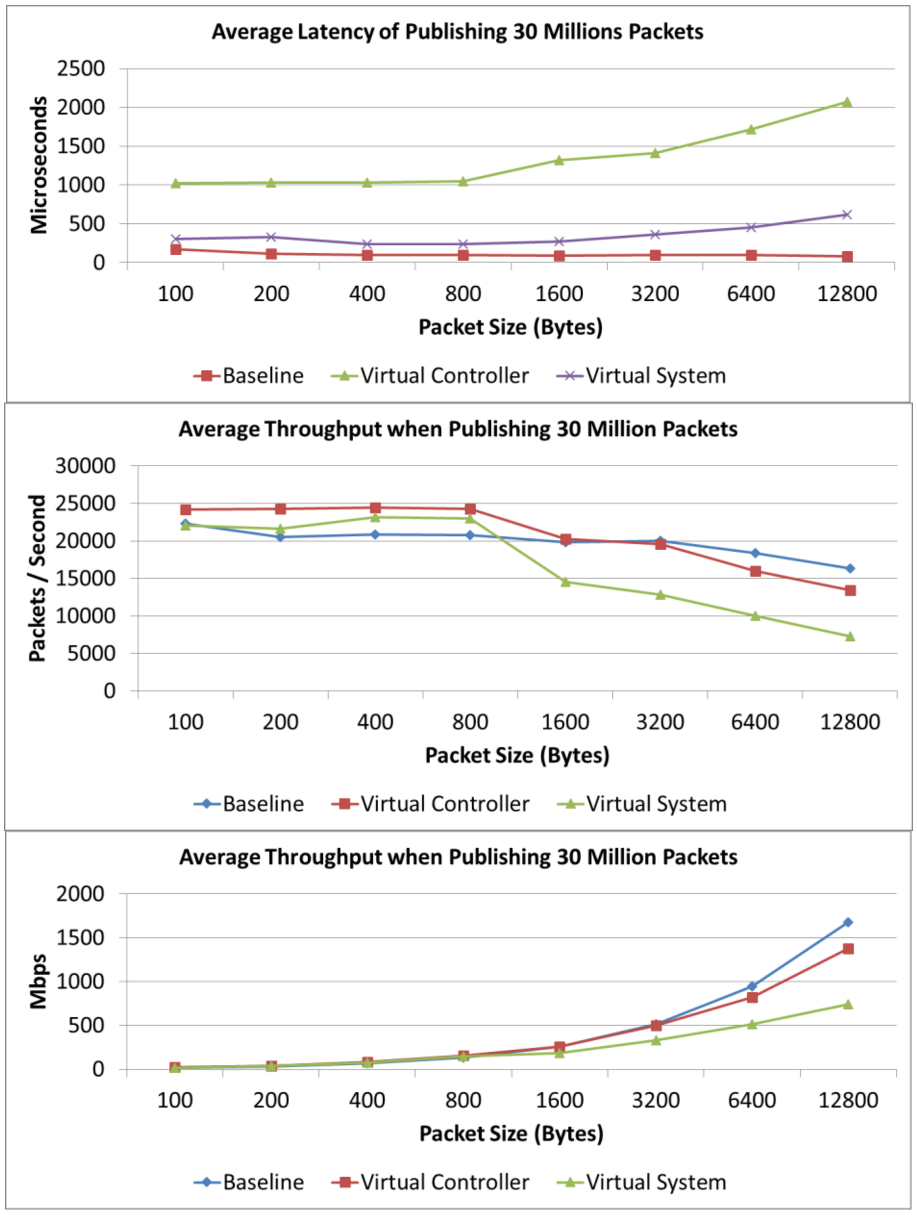


Figure 60 - Average Latency and Throughput Performance Comparison

For packet size of 800 bytes and lower, the virtual controller is slightly higher than the virtual system; but almost 20% higher than the baseline. In summary, the best performance from communication latency and a throughput perspective can be achieved by virtualizing both the controller and I/O system with a maximum packet size of 800 bytes. This is equivalent to a PLC with 3,200 digital I/O signals and 400 analog I/O signals. This conclusion is based on hosting the virtual machines for the controllers and the I/O systems in the same server platform.

5.3.5 Impact Analysis of using Ethernet Switch between the Controller and I/O System in Identical Laptops

The set up for the next performance test configuration is to host the controller application in one laptop and to host the I/O system in another identical laptop. The communication between the controller and the I/O system is implemented through a 16-port 10/100 Mbps 3Com fast Ethernet switch using real-time publish/subscribe DDS middleware. The measured average latency and throughput in terms of packets per second and Megabits per second for the two identical laptops are shown in Figure 61. The communication latency is consistently about 2ms with packet size up to 800 bytes.

The communication latency starts to increase significantly when the packet size increases beyond 800 bytes and would reach to 26ms with packet size of 12,800 bytes. The reason for this increase in communication latency is obvious from the throughput graph where the middleware starts consuming the maximum bandwidth of the Ethernet communication switch of 100 Mbps with packet size of 1,600 bytes. Since the quality of service is set to reliable communication, the middleware starts blocking the packets and throttle the communication with maximum bandwidth available close to 100Mbps. This clearly demonstrates that the throughput is limited by the network capability and not by the CPU or real-time DDS middleware. Although the communication latency is very high with packet size of 12,800 bytes compared to that with packet size of 800 bytes, it is still within the required scan time resolution of 35ms.

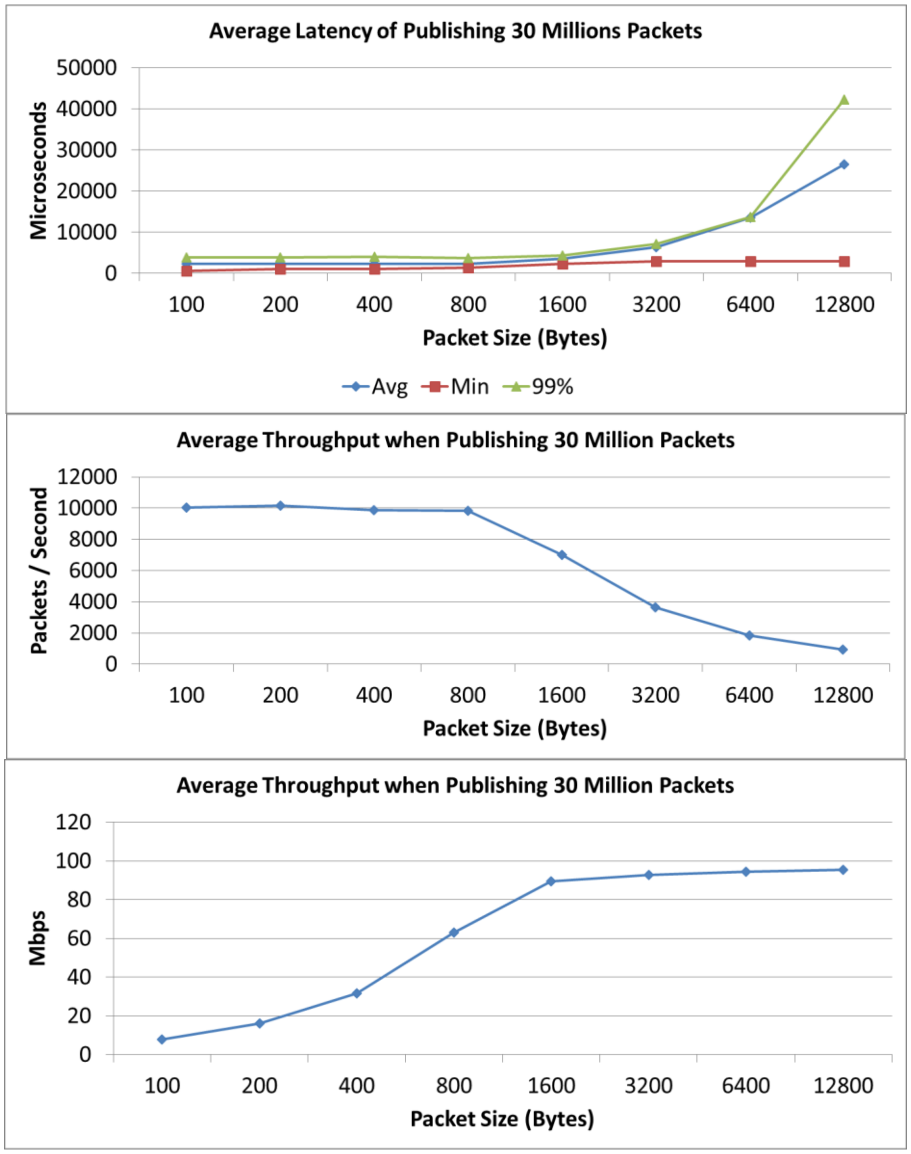


Figure 61 - Average Latency and Throughput Performance Cross 100Mbps Fast Ethernet Switch

5.3.6 Impact Analysis of using Ethernet Switch between the Controller and I/O System in Different Laptops

The next performance test is to demonstrate the impact of having two unequal laptops with different CPU speed. The same configuration is used in the previous test. In the first test, the controller application is hosted in the faster machine. However, in the second test, the controller application is hosted in the slower machine. For the first test, the measured average latency and throughput in terms of packets per second and Megabits per second for the first test are shown in Figure 62.

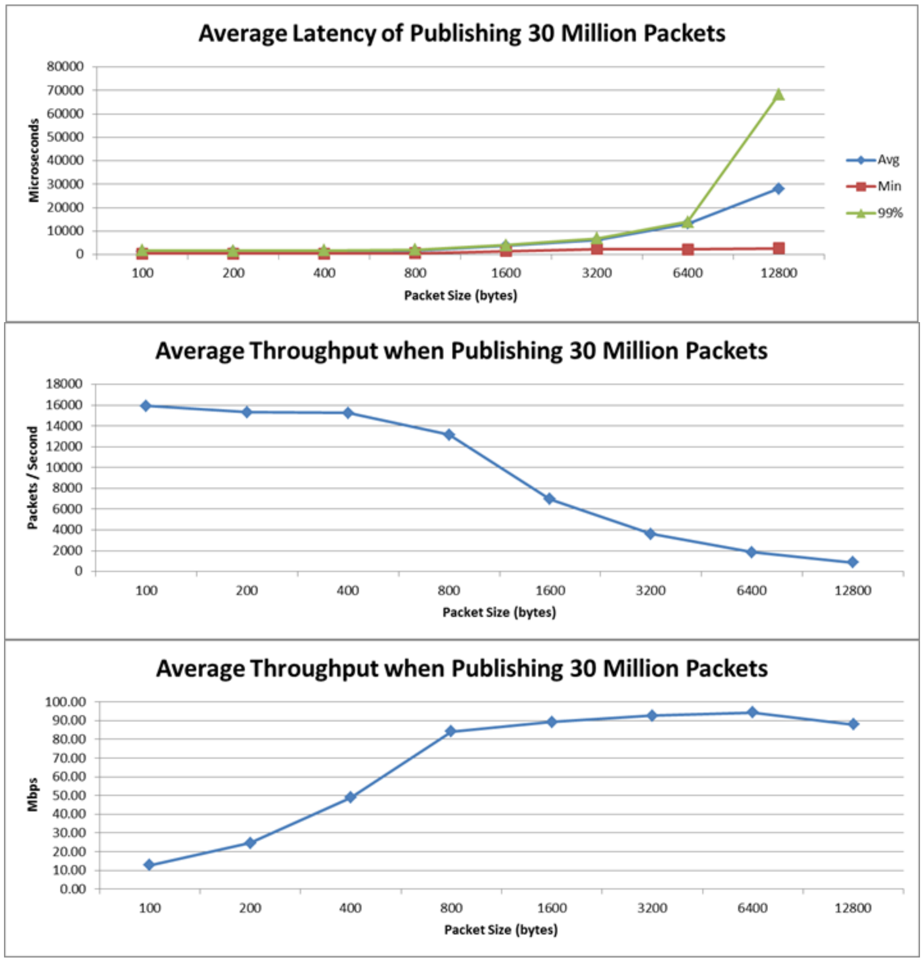


Figure 62 - Average Latency and Throughput Performance with Fast Controller

For the second test, the measured average communication latency and throughput in terms of packets per second and Megabits per second for the first test are shown in Figure 63. The performance is improved when the I/O system is hosted in a faster machine. However, the performance has improved further when the fast machine is used to host the controller application rather than the I/O system.

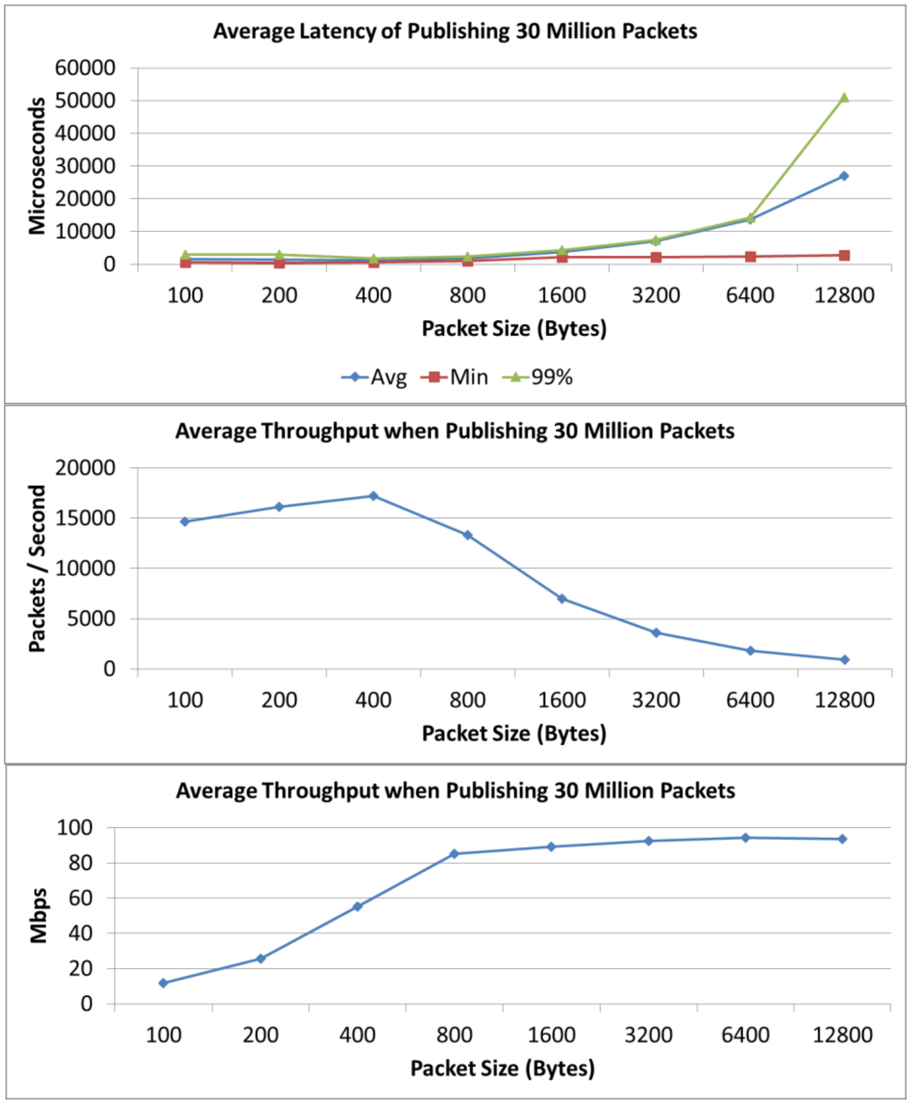


Figure 63 - Average Latency and Throughput Performance with Slow Controller

Because the real-time DDS middleware uses true peer-to-peer messaging with no centralized or message broker, server or daemon processes, it does not impose any inherent limit on the aggregate messaging capacity. It is limited only by the network infrastructure. In all cases, for large systems with packet size beyond 1,600 bytes, it is more efficient to use higher network bandwidth capacity such as the 1 Gbps Ethernet switch.

5.3.7 Communication Latency and Throughput Summary for One Publisher (I/O System) and One Subscriber (Controller)

Figure 64 shows the overall communication latency for one publisher, I/O system, and one subscriber, control application, within each laptop and across 100Mbps Ethernet switch.

There is no significant impact in terms of communication latency due to the middleware

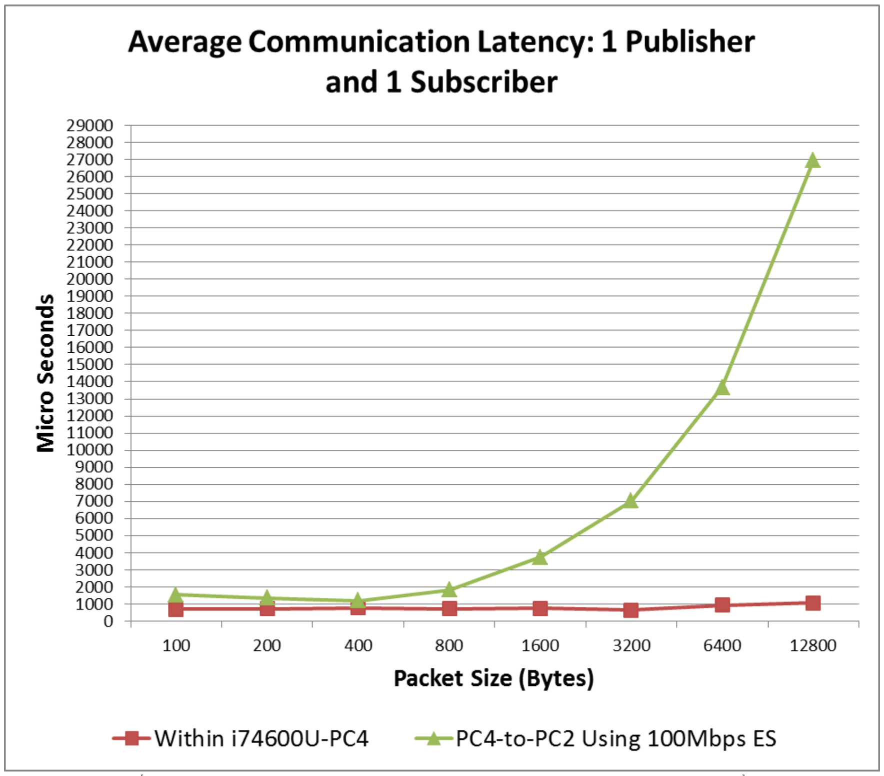


Figure 64 - Average Communication Latency Performance

communication overhead within each laptop. Also, there is a minimum impact due to the Ethernet communication overhead on the performance if the packet size is within 800 bytes. However, with larger packet size beyond 800 bytes, the negative performance impact in communication latency is proportional to the transmission time of the packet through the Ethernet switch. The communication latency within 27ms is adequate for most process automation applications; however, if there is a special need for higher resolution scan time requiring less than 27ms latency, higher bandwidth communication network is recommended. The higher the communication bandwidth, the lower the communication latency that can approach 1ms as demonstrated for the case where both publisher and subscriber applications are within the same computing machine. Figure 65 shows the overall communication throughput for one publisher, I/O system, and one subscriber, control application, within each laptop and across 100Mbps Ethernet switch.

The blue series represent the throughput in terms of packets per second and the red series represent the throughput in terms of Megabits per second. The optimum throughput of 28 thousand packets per second is achieved when the packet size equals 400 bytes where both

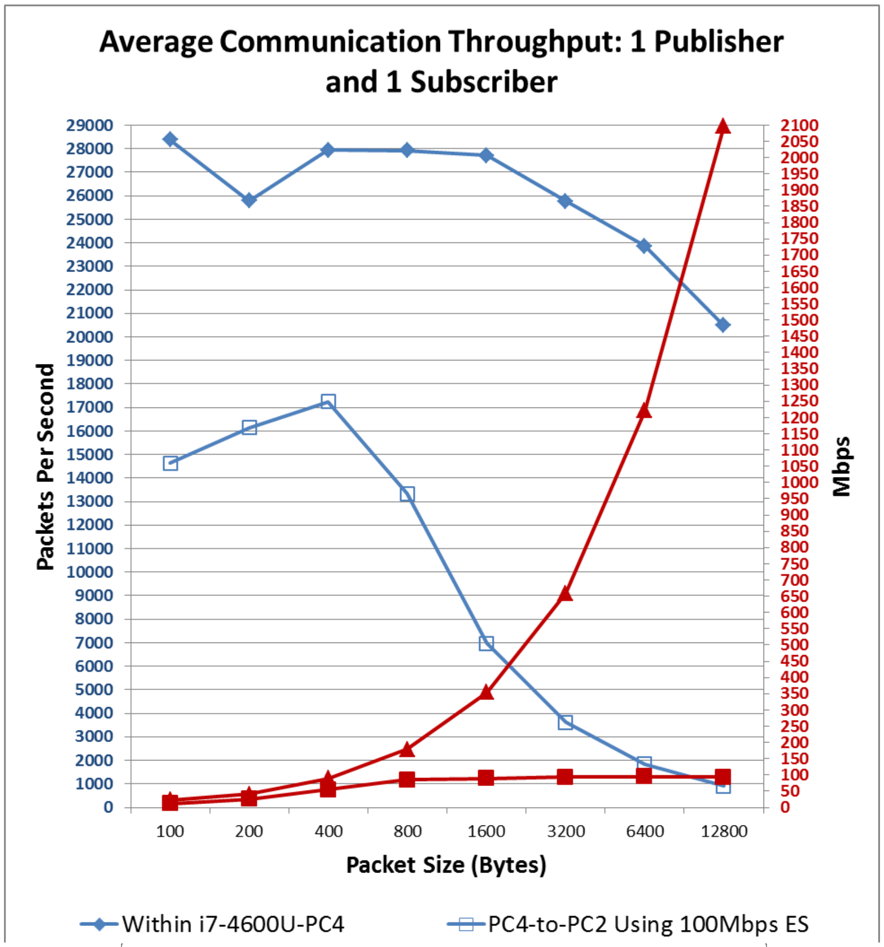


Figure 65 - Average Communication Throughput Performance

publisher and subscriber applications are within the same computing machine. The communication throughput drops down to 20 thousands packets per second when using a packet size of 12800 bytes. However, it is about 18% higher than the optimum result achieved when the communication is crossing the 100Mbps Ethernet switch. Also, the decline slope for the communication throughput, when the packet size is more than 400 bytes, is sharper when the communication between the publisher and subscriber applications is performed through the Ethernet switch. The communication throughput in terms of Megabits per second is calculated by multiplying the throughput rate in terms of packets per second times the size of the packet in bits. The middleware throttles the transmitted packets through the available fast Ethernet communication bandwidth of 100 Mbps.

5.3.8 Performance Test Analysis (One Publisher to Multiple Subscribers Scenarios)

For achieving optimum collaboration among process control applications, the publisher, the I/O system in this case, must provide reliably the right information at the right time to multiple subscribers, which are the associated control applications. Most of the subscribers require the process data information within seconds except for the discrete and regulatory control applications, within 100 milliseconds. To address the worst case scenario, we evaluate the middleware performance in terms of communication latency among all publisher and subscribers to be within 100 milliseconds. Each test scenario is repeated eight times with different data packet sizes of 100, 200 400, 800, 1600, 3200, 6400 and 12800 bytes. The change in data size represents the change in the number of I/O signals. The subscribers measure the throughput by counting the number of received data packets per second and the throughput rate of Megabits per second. The throughput is identical for all subscribers since the real-time middleware is designed with reliable communication quality of service. Therefore, the total throughput is the aggregate throughput at all subscribers. The performance test is to measure the actual communication latency and throughput of multiple control applications and one I/O system scenarios.

5.3.8.1 Communication Latency and Throughput for One Publisher and Multiple Subscribers within One Laptop

The set up for the next performance test configuration is to host one I/O system and multiple control applications, up to four subscribers, in one laptop. The measured average communication latency for four unicast communication scenarios with one, two, three, and four subscribers is shown in Figure 66.

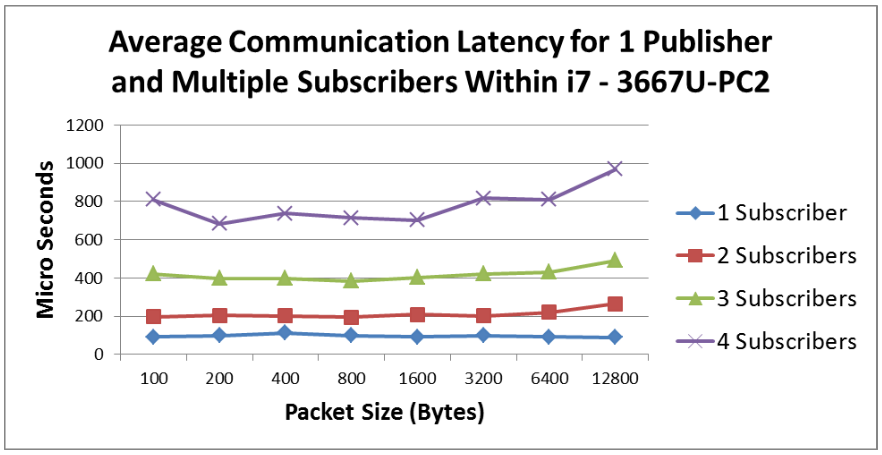
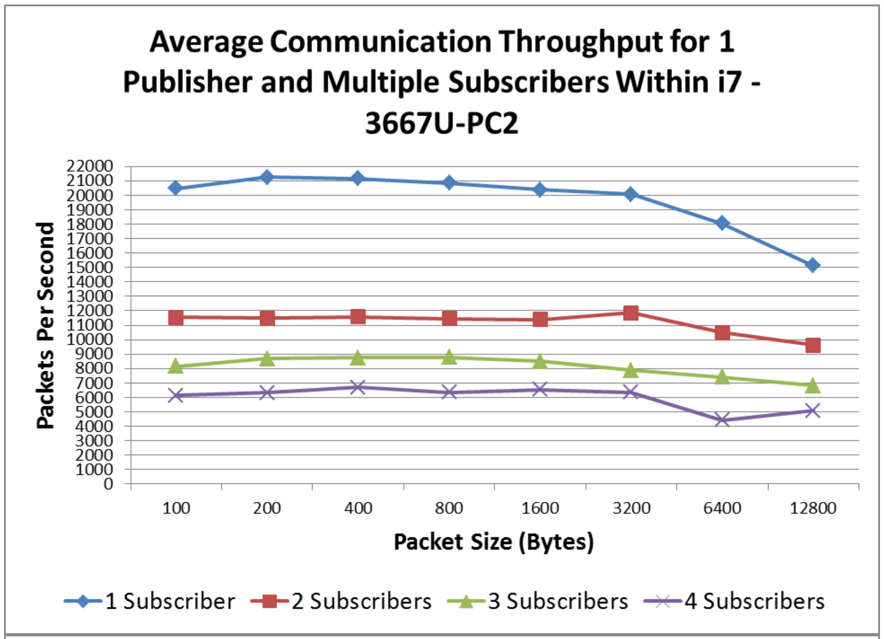


Figure 66 - Average Communication Latency Performance

The performance result of the average communication latency between the multiple control applications and the I/O system is within 1ms, very well below the required scan time resolution while varying the controller size from 100 bytes to a controller size of 12,800 bytes. The data also shows that communication latency remains consistently low as message size increases for the four scenarios. It is noticed that the communication latency is doubled by increasing the number of subscribers by one. In other words, the communication latency is in the range of 100 micro seconds for one subscriber, 200 micro seconds for two subscribers, 400 micro seconds for three subscribers, and in the range of 800 micro seconds for four subscribers. The measured average throughput bandwidth per subscriber between multiple control applications and one I/O system for the fourth laptop measured in packets per second and Megabits per second is shown in Figure 67. The graph shows sustainable publish/subscribe throughput bandwidth. Obviously, the slight decrease in the throughput in terms of number of packets per second is due to the increase in transmission time of each packet. However, the throughput bandwidth in terms of Megabits per second increases significantly with the increase in the size of the packet. This indicates that the real-time DDS middleware was able to cope with the huge increase in data loading and fully utilized available data bus communication bandwidth between the control applications and the I/O system. In other words, it does not impose any inherent limit on the aggregate data messaging capacity, making it suitable for scalable collaborative automation platforms. The highest system throughput recorded in this experiment is 26828 packets per second, for the scenario with one publisher and four subscribers using 400-byte packet size. The lowest system throughput recorded in this experiment is 15128 packets per second, for the scenario with one publisher and one subscriber using 12800-byte packet size.



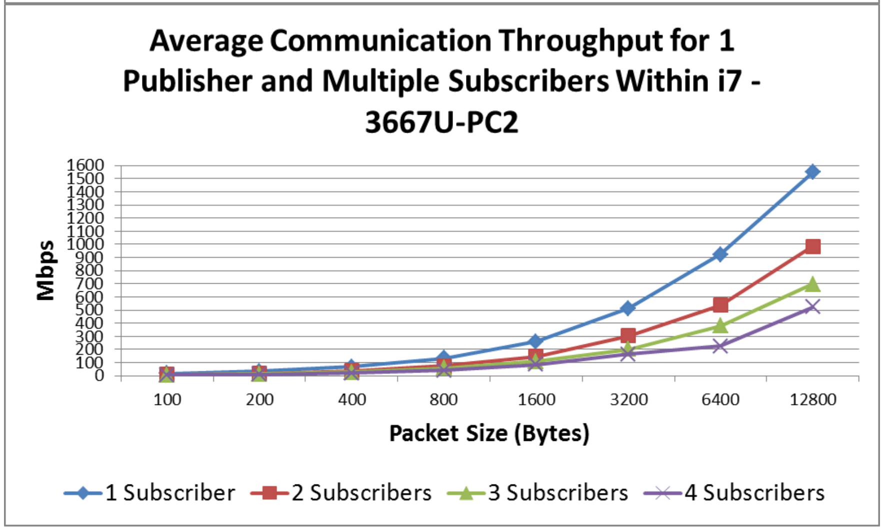


Figure 67 - Average Communication Throughput Performance per Subscriber

The highest system throughput bandwidth recorded in this experiment is 2095 Mbps, for the scenario with one publisher and three subscribers using 12800-byte packet size. The lowest system throughput bandwidth recorded in this experiment is 16.4 Mbps, for the scenario with one publisher and one subscriber using 100-byte packet size.

5.3.8.2 Impact Analysis of using Ethernet Switch between One Publisher and Two Subscribers using Three Laptops

The set up for the next performance test configuration is to host one I/O system in one laptop and two control applications in additional two laptops. The communication among the control applications and the I/O system is implemented through a 16-port 10/100 Mbps 3Com fast Ethernet switch using real-time publish/subscribe DDS middleware. The measured average communication latency using unicast and multicast scenarios compared to the baseline case, hosting the publisher and two subscribers within one laptop, is shown in Figure 68.

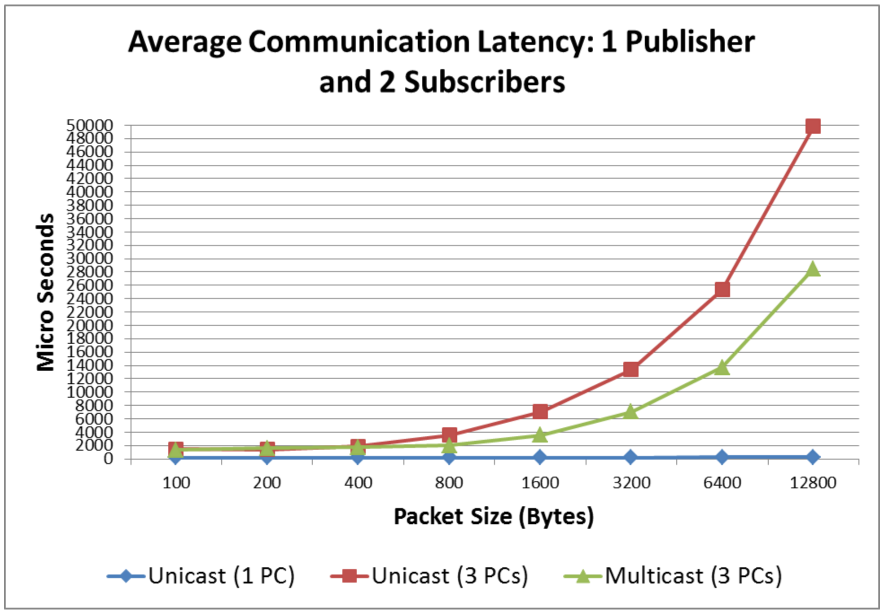


Figure 68 - Average Communication Latency Performance

The performance result of the average communication latency between two control applications and the I/O system for the three cases is within 2ms, very well below the required scan time resolution while varying the controller size from 100 bytes to a controller size of 400 bytes. As the packet size increases beyond 400 bytes, the average communication latency increases significantly using unicast communication with a packet size of 12800 bytes up to 50ms. On the other hand, there is a moderate proportional increase in the average communication latency using multicast communication to the size of the packet and reaches up to 28ms with a packet size of 12800 bytes. Therefore, multicast communication mode is the best method for communicating between a publisher and multiple subscribers in a collaborative automation platform. The measured average communication throughput per subscriber using unicast and multicast scenarios compared to the baseline case, hosting the publisher and two subscribers within one laptop, is shown in Figure 69. The graph shows sustainable publish/subscribe throughput bandwidth between both control applications and the I/O system for the three cases in terms of packets per second while varying the controller size from 100 bytes to a controller size of 400 bytes. Obviously, there is a significant drop in the throughput per subscriber in terms of number of packets per second while increasing the size of the controller beyond 800 bytes due to the increase in transmission time of each packet and the requirement to throttle the communication with maximum bandwidth available close to 100Mbps. However, the multicast communication scenario shows better performance compared to the unicast communication in terms of packets per second. The multicast communication is best suitable for the real-time DDS middleware to cope with the huge increase in data loading and fully utilized available network communication bandwidth between the control applications and the I/O system. In other words, it does not impose any inherent limit on the aggregate data messaging capacity, making it suitable for scalable collaborative automation platforms. It is recommended to sustain the throughput for large collaborative automation platform beyond 400-byte packet size to use higher network bandwidth capacity such as the 1 Gbps Ethernet switch. This will restore the communication throughput performance close to the baseline case as shown in Figure 122 where the baseline throughput is less than 1Gbps for the largest controller using a packet size of 12800 bytes.

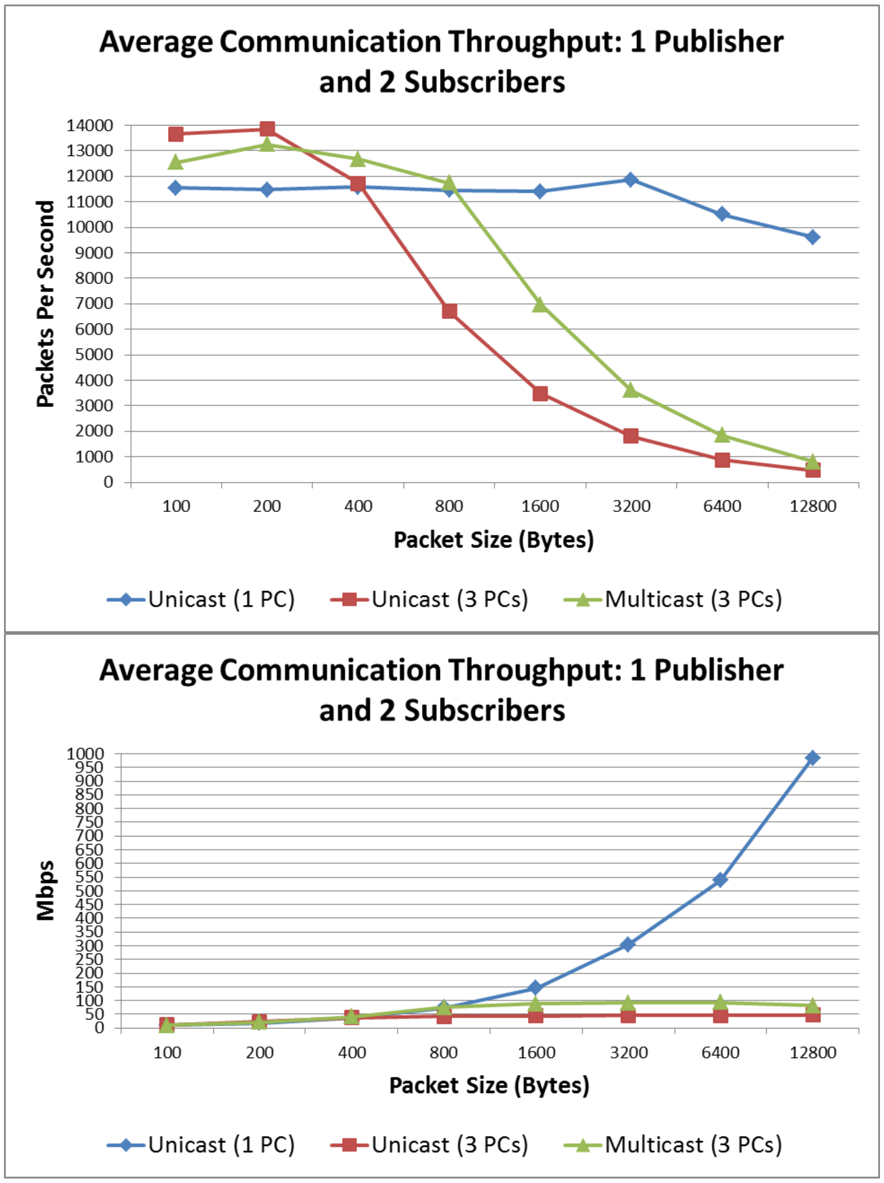


Figure 69 - Average Communication Throughput Performance per Subscriber

5.3.8.3 Impact Analysis of using Ethernet Switch between One Publisher and Multiple Subscribers using Multicast Communication among Three Laptops

The set up for the next performance test configuration is to host one I/O system in one laptop and multiple control applications evenly distributed in additional two laptops. The communication among the control applications and the I/O system is implemented through a 16-port 10/100 Mbps 3Com fast Ethernet switch using real-time publish/subscribe DDS middleware. The measured average communication latency using multicast communication scenarios is shown in Figure 70. The scenarios include two subscribers, four subscribers, six subscribers, eight subscribers, ten subscribers and twelve subscribers.

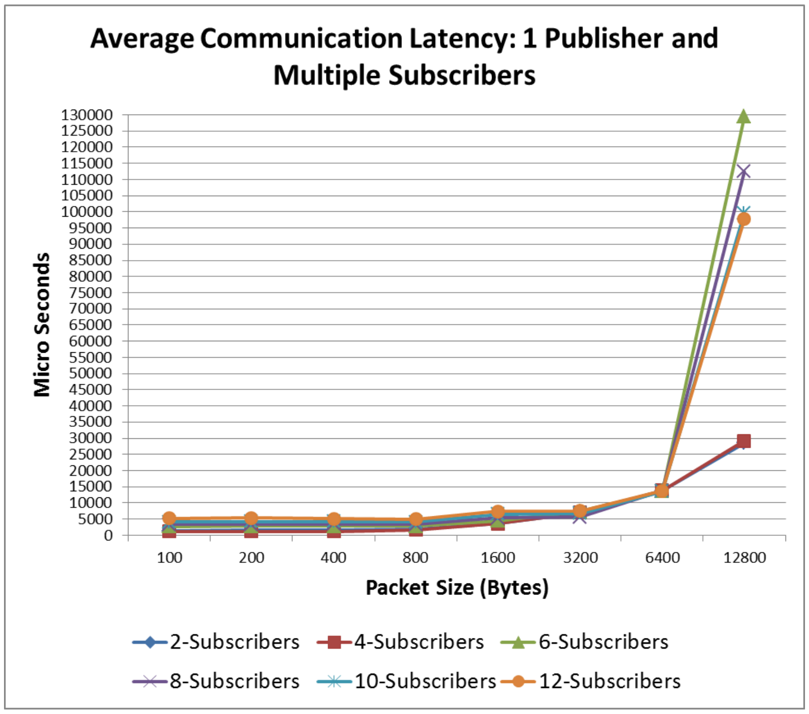


Figure 70 - Average Communication Latency Performance

The performance result of the average communication latency between multiple control applications and the I/O system for the six cases is within 15ms, very well below the required scan time resolution while varying the controller size from 100 bytes to a controller size of 6400 bytes. As the packet size increases beyond 6400 bytes, the average communication latency increases significantly for the cases of six, eight, ten and twelve subscribers with a packet size of 12800 bytes up to 128ms. This is due to the bandwidth limitation within the 100Mbps Ethernet switch. However, for the cases of two and four subscribers, the average communication latency is within 30ms and meeting the required scan time resolution. To reduce communication latency for more than four control applications below 35ms, the control network is required to be upgraded to a 1Gbps Ethernet infrastructure. The measured average communication throughput in packets per second is shown in Figure 71.

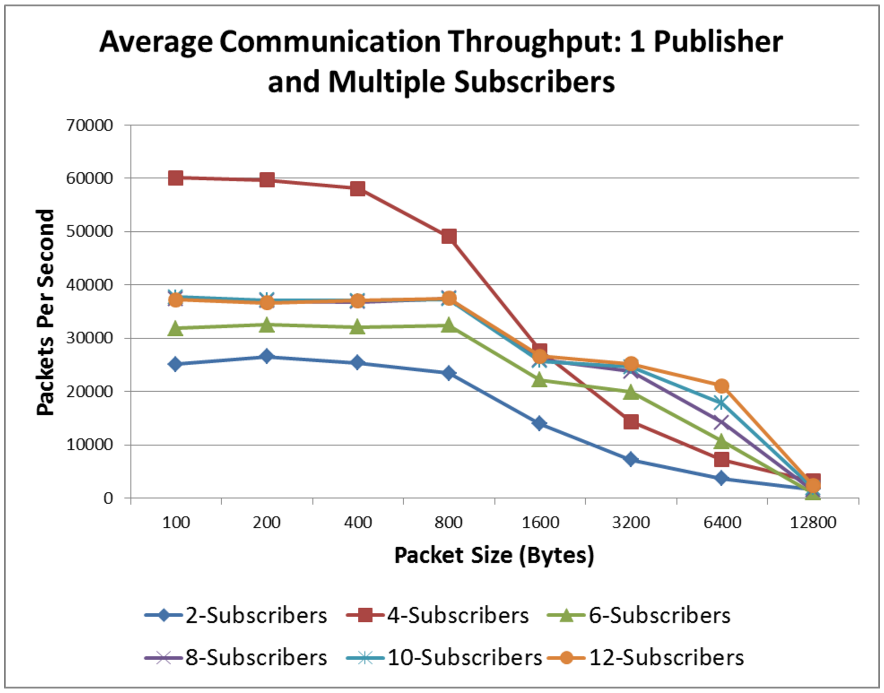


Figure 71 - Average Communication Throughput Performance (Packets/Second)

The graphs show sustainable publish/subscribe throughput bandwidth between the control applications and the I/O system for the six cases in terms of packets per second while varying the controller size from 100 bytes to a controller size of 800 bytes. Obviously, there is a significant drop in the throughput per subscriber in terms of number of packets per second while increasing the size of the controller beyond 800 bytes due to the increase in transmission time of each packet and the requirement to throttle the communication with maximum bandwidth available close to 100Mbps.

Figure 72 shows the measured average communication throughput in Megabits per second.

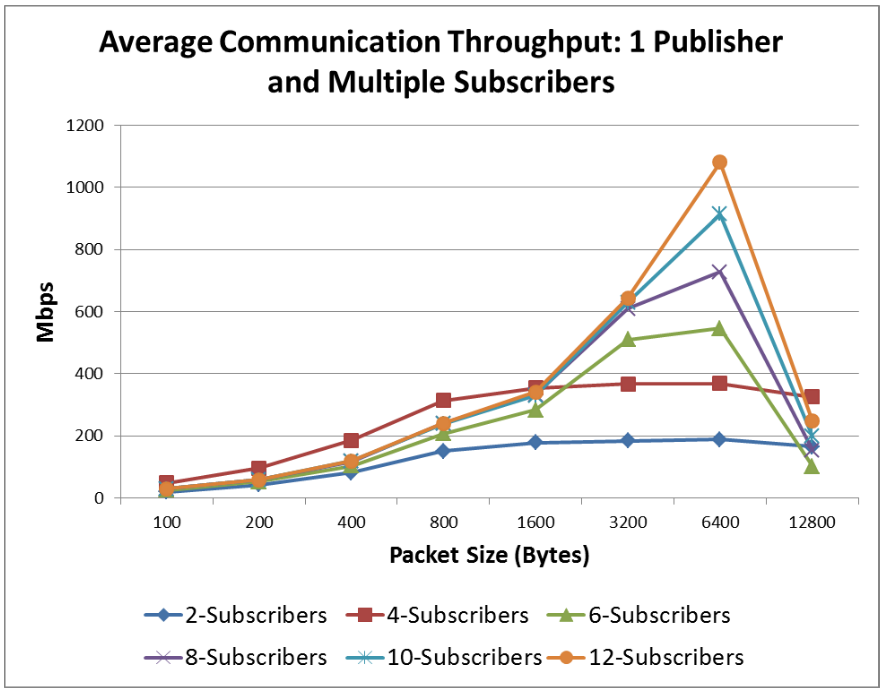


Figure 72 - Average Communication Throughput Performance (Mbps)

The communication throughput increases proportionally to the size of the packet until it reaches the maximum bandwidth available in the Ethernet switch. Each communication link is limited by 100 Mbps. Therefore, for two subscribers, the communication throughput ramps up until it approaches 200 Mbps with packet size of 6400 bytes and starts to drop down after that. Similarly, for twelve subscribers, the communication throughput ramps up until it approaches 1200 Mbps with packet size of 6400 bytes and starts to drop down after that.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In today’s competitive production environment, there is a very high demand on addressing the obsolescence challenges of process automation systems utilized in oil and gas process industries. Extending the life cycle of these systems through incremental upgrades, migration paths, or partial components replacement would lessen the impact of obsolescence challenges to a limited extent and cannot resolve them. The essential strategy enabler in today’s world for safe guarding against premature obsolescence challenges is the utilization of COTS, open source, and/or multi-supplier technologies in order to move from proprietary to standards-based automation solution. To achieve this strategy, this dissertation presented a new ubiquitous data-centric automation and control architecture, called Virtually Clustered Automation Platform, to allow upgrades or replacements of any obsolete components without the need to buy another automation controller altogether. The main concept of this approach is the physical and logical decoupling of the I/O systems from the current proprietary monolithic controllers. However, this change requires real-time reliable and fault tolerant data-centric middleware that provides seamless cross-vendor interoperability. Detailed performance analysis was conducted to evaluate the viability of utilizing the real-time publish/subscribe DDS middleware as a core communication link between the controllers and the I/O systems.

The performance result of the average communication latency between the controller and the I/O system in all tests is very well below the required scan time resolution while varying the controller size from 100 bytes equivalent to a PLC with 400 digital I/O and 50 analog I/O, to a controller size of 12,800 bytes equivalent to a PLC with 80,000 digital I/O and 2,800 analog I/O. Because the real-time publish/subscribe DDS middleware uses true peer-to-peer messaging with no centralized or message broker, server or daemon processes, the performance tests showed that it does not impose any inherent limit on the aggregate messaging capacity, making it suitable for scalable collaborative automation and control platforms.

The following are the advantages of the new ubiquitous data-centric automation and control architecture, vCAP solution:

- It is a cost effective evergreen solution because it is based on field proven interoperable and standard COTS software and hardware components resulting in optimum capital investment for the total cost of ownership throughout the life span of the processing facilities form hurdle to grave.
- It provides an optimum collaborative environment for all control level applications including regulatory control, advanced regulatory control strategies, multivariable advanced control and real-time control optimization.
- It is based on virtual machines that can achieve high utilization since the I/O systems are autonomous and completely independent of the virtual controllers. Adding additional virtual controllers does not require any capital investment for the hardware resulting in maximum total value of ownership.
- It is a high-performance controller because of the decoupling of the I/O systems from the controller and the utilization of virtual networks for segregating the I/O services

from the control activities. The I/O scan update is processed during the control application scan cycle.

- It is highly scalable based on a virtually distributed architecture where I/O modules, CPU and control application are not interwoven. System can grow up by adding multiple virtual machines and/or multiple high-performance fault-tolerant servers for both control and I/O system independently. Also, any changes to the process I/O electrical signals do not have any impact on the control applications.
- It reduces initial capital investment for grass root projects since it does not require any PIBs and their system and marshaling cabinets as well as the associated wiring down to the junction boxes.
- It provides flexibility to capitalize on existing I/O systems of unsupported legacy controllers and provides a cost effective I/O replacement based on failure module by module or channel by channel. This feature can reduce up to 75% of the capital investment for addressing the obsolescence challenges due to obsolete controller, but supported I/O systems.
- It provides centralized security layers of protection with consistent policies cross all servers, similar to the security layers used for cyber security of critical data centers.
- It does not require the actual hardware of the I/O systems during the testing and verification phase at the factory leading to an accelerated project schedule during construction and system installation. Furthermore, testing and troubleshooting the I/O systems is independent from testing and troubleshooting the controllers.

The future work is to extend the vCAP solution to cover the safety control applications in a separate logical DDS domain using the same concept of decoupling the I/O system from the safety controller and using real-time fault-tolerant data-centric middleware for communication among the autonomous safety instrumented interface systems distributed in the safety field junction boxes and the fault-tolerant server-based safety controllers centralized in the collaborative control center. The benefit of this future work is to address the obsolescence challenges of proprietary safety control systems.

References

[1] A. Burns and A. Wellings, “Real-Time Systems and Programming Languages,” 4th Edition, Addison-Wesley, 2009

[2] A. Gokhale, K. Balasubramanian, A. Krishna, J. Balasubramanian, G. Edwards, G. Deng, E. Turkay, J. Parsons, and D. Schmidt, “Model Driven Middleware: A New Paradigm for Developing Distributed Real-Time and Embedded Systems,” Science and Computer Programming, vol. 73, pp. 39-58, 2008

[3] A. González, W. Mata, L. Villaseñor, R. Aquino, J. Simo, M. Chávez and A. Crespo, “µDDS: A Middleware for Real-time Wireless Embedded Systems,” Journal of Intelligent & Robotic Systems, vol. 64, Issues 3-4, pp. 489-503, 2011

[4] A. Kanevsky, A. Skjellum and J. Watts, “Standardization of a Communication Middleware for High-Performance Real-Time Systems, Real-Time Systems Symposium, Paper 168, 1997

[5] A. Krishna and D. Schmidt, “Real-Time CORBA Middleware,” Middleware for Communications, John Wiley and Sons, 2001

[6] A. Krishna, R. Klefstd and D. Schmidt, “Towards Predictable Real-Time Java Object Request Brokers,” 9th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 1-8, 2003

[7] Aberdeen Group, “The Role of Fault-Tolerant Servers in Protecting Virtualized Applications,” Analyst Insight, pp. 1-8, 2013

[8] ARC Market Research Study, “Fieldbus Solutions for Process Industries,” pp. 1-3, 20111

[9] B. Chang, H. Tsai and C. Chen, “Empirical Analysis of Server Consolidation and Desktop Virtualization in Cloud Computing,” Mathematical Problems in Engineering, Article ID 947234, pp. 1-11, 2013

[10] B. Chang, H. Tsai, and C. Chen, “Evaluation of Virtual Machine Performance and Virtualized Consolidation Ratio in Cloud Computing System," Journal of Information Hiding and Multimedia Signal Processing, vol. 4, no. 3, pp. 192–200, 2013

[11] B. Finkemeyer, T. Kröger and F. Wahl, “A Middleware for High-Speed Distributed Real-Time Robotic Applications,” Robotic Systems for Handling and Assembly, STAR 67, pp. 193–212, 2010

[12] B. Galloway and G. Hancke, "Introduction to Industrial Control Networks," IEEE Communications Surveys and Tutorials, vol. 15, issue 2, pp. 860-880, 2013

[13] B. Georges and J. Aubin, "Application of PLC for On-Line Monitoring of Power Transformers," IEEE PES Winter Meeting, vol. 2, pp. 483-486, 2001

[14] B. Gold-Bernstein and W. Ruh, "Enterprise Integration: The Essential Guide to Integration Solutions, Addison Wesley, ISBN-13: 978-0321223906, 2005

[15] B. Liptak, "Process Control, Instrument Engineers' Handbook," 3rd Edition, ISBN-13: 978-0750622554, 1999

[16] B. Lydon, "New Automation Controller Challenges the Status Quo," Automation.com, White Report, pp. 1-16, 2014

[17] B. Polcyn and J. Gustin, "Virtual Fault Tolerant Ethernet Appliance and Method of Operation," US 20130114610 A1 Patent, 2013

[18] B. Reaves and T. Morris, "An Open Virtual Testbed for Industrial Control System Security Research," International Journal of Information Security, vol. 11, pp. 215-229, 2012

[19] C. Bruce-Boye, D. Kazakov, H. Colmorgen, R. Beck, J. Hassan and H. Wojtkowiak, "Middleware-Based Distributed Heterogeneous Simulation," Novel Algorithms and Techniques in Telecommunications and Networking, pp. 333-337, 2010

[20] C. Murthy and G. Manimaran, "Resource Management in Real-Time Systems and Networks," The MIT Press, ISBN-13: 978-0262133760, 2001

[21] D. Coughanowr and S. LeBlanc, "Process Systems Analysis and Control," 3rd Edition, ISBN-13: 978-0073397894, 2009

[22] D. Hill and H. Forbes, "ExxonMobil and Lockheed Martin – 20 Questions about Open Automation," ARC Report, pp. 1-8, 2016

[23] D. Schmidt and F. Kuhns, "An Overview of the Real-Time CORBA Specification," IEEE Computer Society, vol. 33, no. 6, pp. 56-63, 2000

[24] D. Wallance, "How to Put SCADA on the Internet," Control Engineering, vol. 50, no. 9, pp. 16-21, 2003

[25] E. Aksoy, S. Canbek and N. Adar, " DDS-Based Heterogeneous Robots Communication Middleware," Computer Technology, vol. 3, no, 1, Special Issue, pp. 95-100, 2011

[26] E. Ozdemir and M. Karacor, “Mobile Phone Based SCADA for Industrial Automation,” ISA Transactions, vol. 45, pp. 67-75, 2006

[27] E. Parr Industrial, “Control Handbook,” Industrial Press Inc., ISBN-9780849385704, 1999

[28] F. Guthrie, S. Lowe and M. Saidel-Keesing, “VMware vSphere Design,” John Wiley & Sons, ISBN-13: 978-1118407912, 2011

[29] F. Jammes and H. Smith, “Service-Oriented Paradigms in Industrial Automation,” IEEE Transactions Industrial Informatics, vol. 1, no. 1, pp. 62-70, 2005

[30] F. Plášil and M. Stal, “An Architectural View of Distributed Objects and Components in CORBA, Java RMI and COM/DCOM,” Software-Concepts and Tools, vol. 19, no. 1, pp. 14-28, 1998

[31] G. Bernat, A. Burns and A. Llamosi, “Weakly Hard Real-Time Systems,” IEEE Transactions on Computers, vol. 50, no. 4, pp. 308-321, 2001

[32] G. Buttazzo, “ Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications,” Springer, ISBN 978-1-4614-0676-1, 2011

[33] G. Coulouris, “Distributed Systems: Concepts and Design,” 5th Edition, Addison-Wesley, ISBN-13: 978-0132143011, 2011

[34] G. McMillan and D. Considine, “Process/Industrial Instruments and Controls Handbook,” 5th Edition, McGraw-Hill, ISBN-13: 063-9785313175, 1999

[35] G. Pardo-Castellote, “ Data-Centric Programming Best Practices: Using DDS to Integrate Real-World Systems,” Real-Time Innovations, Inc., pp. 1-18, 2010

[36] G. Pardo-Castellote, “ OMG Data-Distribution Service: Architectural Overview,” Real-Time Innovations, Inc., pp. 1-7, 2005

[37] G. Wells and A. Chalmers, “A Comparison of Linda Implementations in Java,” Communicating Process Architectures, vol. 58, pp. 63-75, 2000

[38] G. Xiaohui, W. Yu, W. Beibei and Liyongqing, “A New Type of Center Data Structure in Cloud Computing,” Information Technology Journal, vol. 13, no. 3, pp. 461-468, 2014

[39] H. Duran-Limon and G. Blair, “Adaptive Resource Management in Middleware: A survey,” IEEE Computer Society, vol. 5, no. 7, 2004

[40] H. Perez and J. Gutierrez, “A Survey on Standards for Real-Time Distribution Middleware,” ACM Computing Surveys, vol. 46, no. 4, Article 49, pp. 1-39, 2014

[41] I. Calvo, F. Perez, I. Etxeberria-Agiriano and O. Garcia de Albeniz, “Designing High Performance Factory Automation Applications on Top of DDS,” International Journal of Advanced Robotic Systems, vol. 10, no. 205, pp. 1-12, 2013

[42] J. Al-Jaroodi and N. Mohamed, “Middleware Trends for Network Applications,” Journal of Network and Computer Applications, vol. 33, pp. 523–524, 2010

[43] J. Al-Jaroodi and N. Mohamed, “Service-Oriented Middleware: A survey ,” Journal of Network and Computer Applications, vol. 35, pp. 211–220, 2012

[44] J. Buck, S. Ha, E. Lee, D. Messerschmitt, “ Ptolemy: A Framework for Simulating and Prototyping Heterogeneous Systems,” International Journal of Computer Simulation, pp. 1-34, 1992

[45] J. Katzel, “Information Systems: The Evolution of the HMI,” Control Engineering, pp. 1-8, 2012

[46] K. An, A. Gokhale, D. Schmidt, S. Tambe, P. Pazandak and G. Pardo-Castellote, “Content-based Filtering Discovery Protocol (CFDP): Scalable and Efficient OMG DDS Discovery Protocol,“ DEBS ’14, pp. 1-12, 2014

[47] K. Hayashi, “Reliability Improvement is Key to Server Virtualization, Fault-Tolerant Servers Get Another Look,” NEC White Paper, pp. 1-9, 2013

[48] K. Kim and P. Kumar, “Design and Experimental Verification of Real-Time Mechanisms for Middleware for Networked Control,” American Control Conference, pp. 2119-2124, 2010

[49] K. Ramamritham and S. Son, “Real-time databases and data services,” Real-Time Systems, vol. 28, no. 2, pp. 179-215, 2004

[50] L. Gong and C. Kulikowski, “An Intelligent Groupware Environment for Real-Time Distributed Medical Collaboration,” The 21st symposium on Computer Applications in Medical Care, American Medical Informatics Association , pp. 959, 1997

[51] L. Zou, Z. Wang, H. Dong, Y. Liu and H. Gao, “Time- and Event-Driven Communication Process for Networked Control Systems: A Survey,” Hindawi Publishing Corporation, vol. 2014, Article ID 261738, pp. 1-10, 2014

[52] M. Anand, S. Sarkar and S. Rajendra, “Application of Distributed Control System in Automation of Process Industries,” International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 6, pp. 377-383, 2012

[53] M. Henning, “ A New Approach to Object-Oriented Middleware,” IEEE Internet Computing, vol. 8, no. 1, pp. 66-75, 2004

[54] M. Hossain and D. Semere, “ Virtual Control System Development Platform with the Application of PLC Device,” International Multi Conference of Engineers and Computer Scientists, vol. 2, pp. 1-6, 2013

[55] M. Laughton and D. Warne, “Electrical Engineers' Reference Book,” Newnes, 16th Edition, ISBN-13: 978-0750646376, 2002

[56] M. Maher, “Real-Time Control and Communications,” 18th Annual ESD/SMI International Programmable Controllers Conference Proceedings, pp. 431-436, 1989

[57] M. Mahmoud and Y. Xia, “Analysis and Synthesis of Fault-Tolerant Control Systems,” Wiley, ISBN-13: 978-1118541333, 2014

[58] M. Musial, V. Remu and G. Hommel, “Middleware for Distributed Embedded Real-Time Systems,” Embedded Systems Modeling – Technology, and Applications, pp. 111–120, 2006

[59] M. Valls, I. López and L. Villar, “iLAND: An Enhanced Middleware for Real-Time Reconfiguration of Service Oriented Distributed Real-Time Systems,” IEEE Transactions on Industrial Informatics, vol. 9, no. 1, pp. 228-236, 2013

[60] N. Mahalik, “Attributes of Industrial Machine and Process Control Systems,” International Journal of Computer Applications in Technology, vol. 25, no. 4, pp. 234-240, 2006

[61] N. Medvidovic, “The Role of Middleware in Architecture-Based Software Development,” International Journal of Software Engineering and Knowledge Engineering, vol. 13, no. 4, pp. 367-393, 2003

[62] Object Management Group, “Data Distribution Service for Real-Time Systems Specification,” Version 1.4, 2015

[63] Object Management Group, “the Real-Time Publish-Subscribe Protocol (RTPS) DDS Interoperability Wire Protocol Specification,” Version 2.2, 2014

[64] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt and A. Warfield, "Xen and the Art of Virtualization," 19th ACM Symposium on Operating Systems Principles, pp. 164-177, 2003

[65] P. Friedmann and T. Stoltenberg, "Continuous Process Control," Instrument Society of America, ISBN – 9781556175572, 1996

[66] P. Wyckoff and S. McLaughry, "T Spaces," IBM Systems Journal, vol. 37, no.3, pp. 454-474, 1998

[67] Q. Li, "The Future-Oriented Middleware Technology," Journal of Computers, vol. 5, no. 2, pp. 250-257, 2010

[68] Q. Liang and L. Li, "The Study of Soft PLC Running System," Advanced in Control Engineering and Information Science, vol. 15, pp. 1234-1238, 2011

[69] R. Starke and R. de Oliveira, "System-Management-Mode in Real-Time PC-Based Control Applications," Journal of Control Automation Electrical Systems, vol. 24, pp. 430-438, 2013

[70] S. Oh, J. Kim and G. Fox, "Real-Time Performance Analysis for Publish/Subscribe Systems," Future Generation Computer Systems, vol. 26, no. 3, pp.318-323, 2010

[71] S. Paul, J. Pan and R. Jain, "Architectures for the Future Networks and the Next Generation Internet: A Survey," Department of Computer Science and Engineering- Washington University in St. Louis, pp. 1-59, 2009

[72] S. Ragavan, V. Ganapathy and I. Kusnanto, "Rapid Automation Application Deployment Framework for Real-Time Process and Industrial Automation Systems," International Journal of Computer Theory and Engineering, vol. 6, no. 6, pp. 515-520, 2014

[73] V. Trevathan, "A Guide to the Automation Body of Knowledge, 2nd Edition, ISA-Instrumentation, Systems & Automation Society, ISBN-13: 978-1556179846, 2006

[74] W. Bolton, "Programmable Logic Controllers," 5th Edition, Newnes, ISBN-13: 978-1856177511, 2005

[75] W. Heinzelman, A. Murphy, H. Carvalho and M. Perillo, " Middleware to Support Sensor Network Applications," IEEE Network Magazine, vol. 18, 2004

[76] W. Kang, K. Kapitanova and S. Son, "RDDS: A Real-Time Data Distribution Service for Cyber-Physical Systems," IEEE Transactions on Industrial Informatics, vol. 8, no. 2, pp. 393-405, 2012

[77] W. Levine, “The Control Handbook,” CRC Press, 2nd Edition, BN-13: 978-1420073669, 2010

[78] W. Xiao, H. Liang and B. Parhami, “ A Class of Data-Center Network Models Offering Symmetry, Scalability, and Reliability,” Parallel Processing Letters, vol. 22, no. 4, pp. 1-10, 2012

[79] Z. Ma, Z. Sheng and L. Gu, “ DVM: A Big Virtual Machine for Cloud Computing,” IEEE Transactions on Computers, vol. 63, no. 9, pp. 2245-2258, 2014

Vitae

Name	: Ghalib A. Al-Hashim
Nationality	: Saudi
Date of Birth	: 10/7/1965
Email	: ghalib.hashim@gmail.com
Address	: Dhahran, P.O. Box 9502
Academic Background	: Computer Engineering
Published Papers	: "Novel Design of Ubiquitous Data-Centric Automation and Control Architecture", ISA process Control & Safety Symposium, November 2015
	"Novel Design of Collaborative Automation Platform Using Real-Time DDS for an Optimum Control Environment", Journal of Circuits, Systems, and Computers, February 2016
	"Novel Design of Heterogeneous Automation Controller Based on Real-Time DDS to Avoid Obsolescence Challenges", Journal of Circuits, Systems, and Computers, April 2016