ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

An Off-line Simulation Test and Verification System for Third-party Communication Stations of Nuclear Power Plant DCS

Zhaoxu Kang[1, a]

[1]Liaoning Hongyanhe Nuclear Power Co., Ltd., Dalian 116300, China

[a]lovecaf@163.com

Abstract

The distributed control system (DCS) of the nuclear power plant has communication interfaces with many third-party systems, such as APA, RGL, GRE/GME, LSS, SEK and JDT. Before replacing a faulty communication station, you have to do parameter configuration, Linux installation, serial port and protocol configuration, but there is no verification window or off-line platform, so a spare station may fail to establish communication on site. Therefore, an off-line simulation test and verification system based on Python and virtualization technology is designed and implemented. The system consists of a simulation server, an engineer station, communication stations, a KVM switch and a movable cabinet. The simulation server runs CentOS and simulation software for MODBUS-TCP, IEC 60870-5-104 and MODBUS-RTU; The engineer station uses VMware to reproduce Windows XP/7 configuration environments; The station establishes a real data link with the simulation server. Application results show that the system can fully verifies link establishment, functional intact, point-value consistency before on-site deployment. What's more, eliminating the risk of unverifiable spare stations and improving equipment reliability.

Keywords

Distributed control system (DCS); communication station; protocol simulation; off-line verification; virtualization.

1. INTRODUCTION

The distributed control system (DCS) is the core of the instrumentation and control (I&C) system of a nuclear power plant. It takes charge in the signal acquisition, monitoring and control of unit process, so its reliability is directly related to the safe and stable operation of the unit. The DCS of a nuclear power plant does not operate in isolation. It has communication interfaces with many third-party systems, such as APA, RGL, GRE/GME, LSS, SEK and JDT. A prominent contradiction is that there is no window during unit operation to perform on-line inspection of the replaced communication station, and there is no off-line platform capable of functional verification in advance, so after a spare part is applied to the site there is a risk that communication cannot be established. For a nuclear power plant that places equipment reliability first, such a situation of "first use upon deployment of the spare" is unacceptable.

This paper designs and implements a communication simulation test and verification system based on the Python programming language[5] and virtual machine technology. Its core idea is to build a movable communication simulation test environment at the nuclear power plant site, in which a protocol server-side simulation program written in Python acts as the "virtual opposite end" of the communication station, and virtualization technology reproduces the engineer station configuration environment, so that the communication station under

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

verification establishes a real data communication link under the same configuration, point table and protocol conditions as on site, thereby completing full functional verification before being applied to the site. The main contributions of this paper include: (1) proposing the overall scheme of an off-line simulation test and verification system for third-party communication stations of nuclear power plant DCS, which integrates protocol server-side simulation, a virtualized configuration environment and the real device under test into a mobile cabinet; (2) developing server-side simulation software for the three protocols MODBUS-TCP, IEC 60870-5-104[3][4] and MODBUS-RTU[2], with a built-in random-variation function of communication points to simulate dynamic data; (3) using VMware virtualization technology to reproduce two configuration environments, Windows XP and Windows 7, therefore, realizing the real downloading of communication point tables and configuration files; (4) establishing a five-step verification process covering power-on inspection, log inspection, link diagnosis, point-value comparison and telegram byte-level comparison, and applying it in the spare-part verification of an actual unit.

2. SYSTEM SCHEME

2.1. Overall Architecture

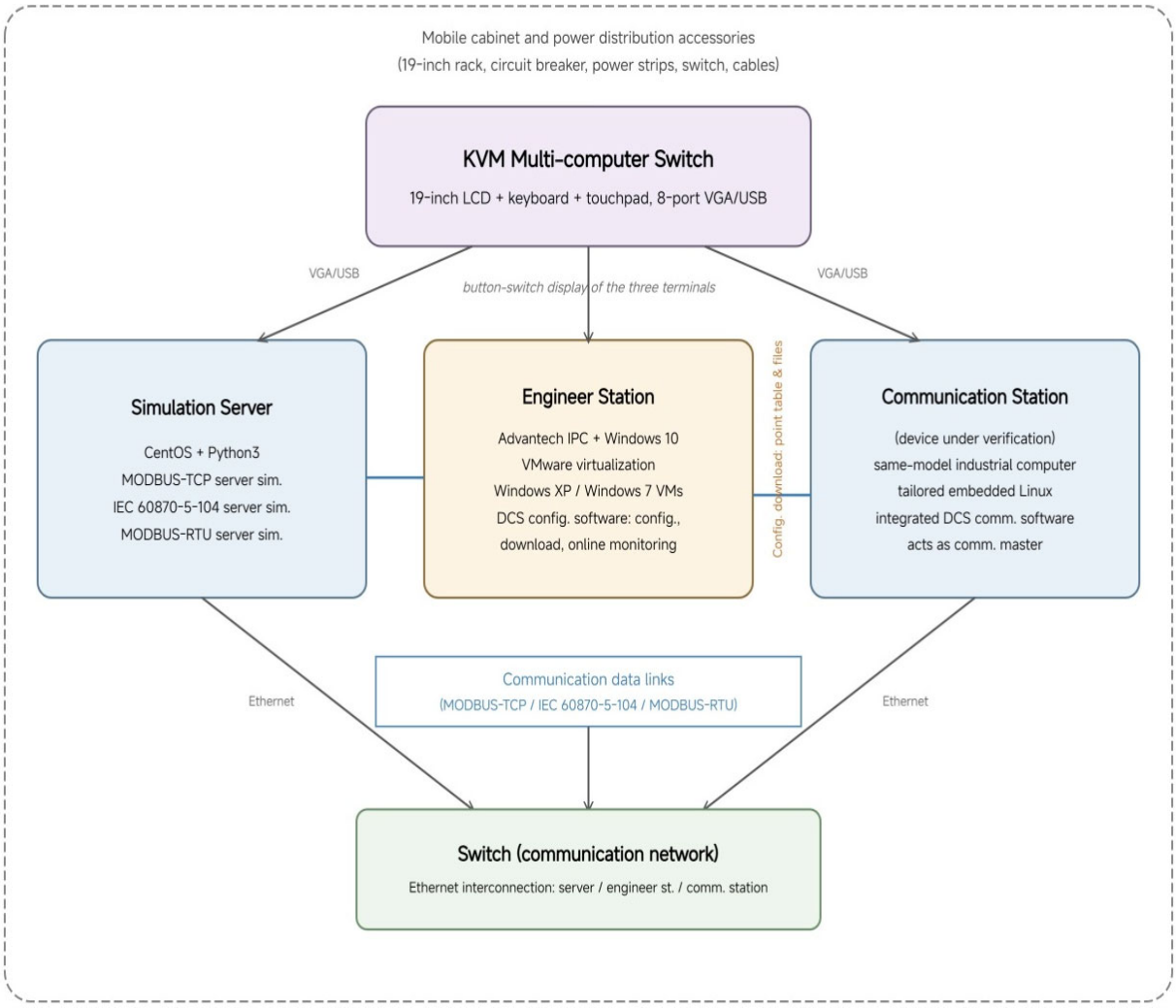


Figure 1. Structure diagram of the communication simulation test and verification system

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

The design goal of the verification system is to provide the communication station under test with a communication opposite end and a configuration environment consistent with the site without changing any of its software or hardware configuration, so that its communication function can be fully and truly examined. Figure 1 shows that the overall architecture of the system consists of five parts: a KVM multi-computer switch, a simulation server, an engineer station, communication stations, and a movable cabinet with power distribution accessories. The simulation server runs the CentOS operating system and the server-side simulation software written by Python to respond to the data requests of the communication station; the engineer station runs Windows 10 and uses VMware virtualization technology to emulate the two operating systems Windows XP/7, realizing DCS configuration and downloading; the communication station uses an industrial computer of the same model as on site, installed with the tailored embedded Linux and communication software which can establishes communication with the simulation server as the communication master;

2.2. Design of the Protocol Server-Side Simulation Software

The simulation server is the core software carrier of the system. It runs the CentOS operating system and the Python 3 programming environment with server-side simulation code. After the code runs, it can truly and completely simulate the protocol server side, respond normally to the data requests of the communication station, and form a complete data communication link together with the communication station.

The simulation software has a built-in random-variation program for communication points, which can simulate random changes of communication point values during verification. It can examining the acquisition and forwarding capability of the communication station for dynamic data and avoiding the problem that "false communication" is difficult to expose under static data. The server-side programs of the three protocols are deployed as independent Python scripts; During verification, the corresponding test is started by a terminal command with different scripts.

2.3. Virtualization-Based Engineer Station Configuration Environment

The engineer station undertakes the installation and running of the DCS configuration software. Since the configuration software of different on-site systems relies on the Windows XP and Windows 7 operating systems respectively, satisfying both environments directly on a physical machine is neither economical nor reliable. Therefore, the engineer station is configured with an Advantech industrial computer, with Windows 10 installed as the host, and a virtual environment with VMware software [6] to emulate the two operating systems Windows XP/7 and installing the corresponding versions of the DCS configuration software respectively. The virtual machines connect with the communication station through bridged networking, so that the virtual machine configuration environment is equivalent to an independent physical machine on the communication link, thereby realizing the real configuration and downloading of communication point tables and communication configuration files as well as the engineering online monitoring function. This design enables a single engineer station to cover all on-site configuration software environments, significantly reducing the hardware complexity of the verification system.

2.4. Hardware Integration and Implementation

As shown in Figure 2,to meet the requirement of flexible deployment at the nuclear power plant site, all equipment of the system is integrated into a movable cabinet.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

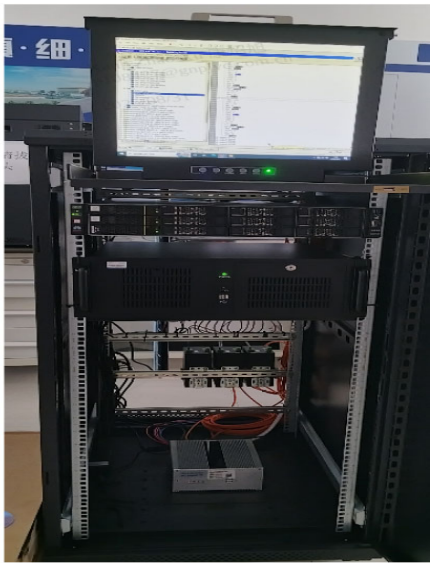


Figure 2. Physical view of the communication simulation test and verification system

3. ANALYSIS AND VERIFICATION

3.1. Verification Object and Verification Process

The verification objects are same-model communication stations taken from the warehouse and intended to be replaced for on-site use. The preparatory work before verification includes: checking the simulation server, engineer station and communication station are all powered on normally; Switching to the simulation server interface through the KVM, logging in and starting the server-side simulation program of the corresponding protocol with the python3 command in the terminal; Switching to the engineer station, logging into the corresponding virtual machine system and downloading the communication point table and configuration files. The communication station itself supports three communication protocols; It starts sending data request telegrams as the communication master after different communication point tables and configuration files are downloaded by the engineer station.

3.2. Verification Results and Data Analysis

The methods, criteria and results of each verification item are shown in Table 1.

The results show that under all three protocols, the communication station can be powered on normally and enter the system interface (S01); the communication logs confirm that the communication software starts normally (S02); and the communication diagnostic program confirms that the communication link between the communication station and the simulation server is established normally (S03). In the engineering online step (S04), the built-in random-variation program of communication points on the simulation server side simulates random point-value changes, and the corresponding communication address point values read in real time by the engineer station change normally and are completely consistent with the point values on the simulation server side; Figure 4(a) shows that the variation curves of the set value on the simulation server side and the online reading value of the engineer station for a MODBUS-TCP holding register point within a 120 s verification window; the two curves completely coincide, and the point-value data at the sampling instants are shown in Table 2. As shown in Figure 4(b) the point-value consistency rates of the five verification rounds of each of the three protocols are all 100%. In the telegram recording step (S05), the communication telegram recording function of the communication station is enabled to record communication telegrams in real time; taking the MODBUS protocol as an example, the byte information in the telegram (slave address, function code, start address, data length, check code, etc.) is printed in

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

detail and compared byte by byte with the communication point table configured in the engineer station, and all fields are consistent with the configuration.

Table 1. Functional verification items and results of the three communication protocols

Verification item	Verification method	Criterion	MODBUS-TCP	IEC 60870-5-104	MODBUS-RTU
Power-on and system startup	Switch display to check the system interface (S01)	Enters the system normally	Pass	Pass	Pass
Communication software startup	Check communication logs (S02)	No abnormality in logs	Pass	Pass	Pass
Link establishment	Communication diagnostic program (S03)	Link established normally	Pass	Pass	Pass
Point-value consistency	Engineering online comparison of point values on both sides (S04)	equal	Pass	Pass	Pass
Telegram byte comparison	Telegram records checked byte by byte against the point table (S05)	All fields consistent with the configuration	Pass	Pass	Pass

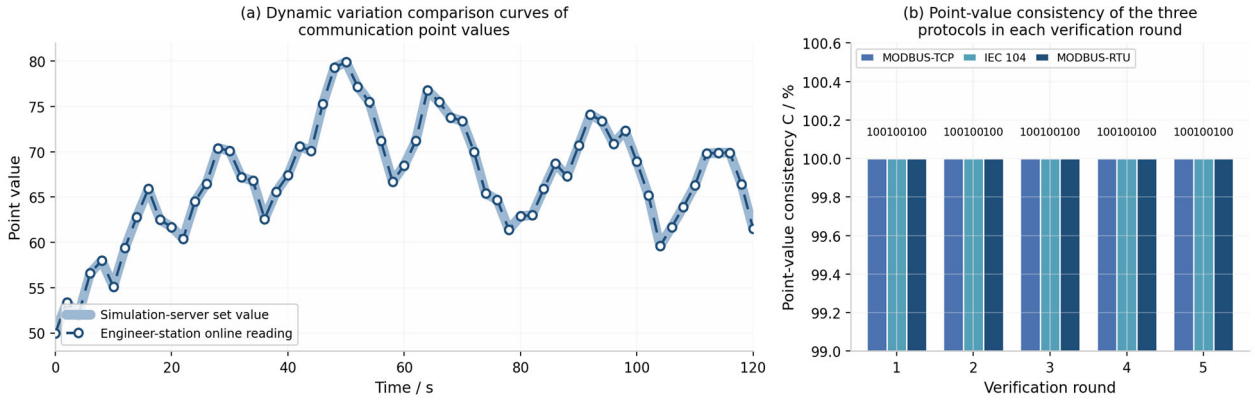


Figure 4. Dynamic point-value verification curves and consistency statistics of communication points

Table 2. Point-value data at the sampling instants of the curves in Fig. 4(a)

Sampling instant t/s	0	30	60	90	120
Point value on simulation server side	50.0	70.1	68.5	70.7	61.5
Point value on engineer station side	50.0	70.1	68.5	70.7	61.5

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

The verification results of all three protocols passed. This result shows that: on the one hand, the server-side simulation software written in Python truly and effectively simulates the standard response formats of the three protocols, and the communication station cannot distinguish the simulated opposite end from the real on-site opposite end; on the other hand, the five-step verification process forms progressively deepening criteria from "link connected" to "data accurate" and then to "telegram correct", and any abnormality at any layer can be located independently, facilitating rapid troubleshooting of link parameters, system installation, serial port or protocol configuration problems.

3.3. Analysis of Application

The system has been put into application at the nuclear power plant site for the functional verification of spare communication stations. The application value is reflected in three aspects. First, the verification gate is moved forward: the communication station completes full functional verification before being applied to the site, eliminating the potential risk that communication cannot be established after a spare part is applied to the site and improving equipment reliability. Second, the link is real and credible: unlike general-purpose protocol test tools, in this system the communication station runs the real tailored Linux and communication software and loads the real communication point table and configuration files, and the configuration environment is also consistent with the site, so the verification conclusion can be directly extrapolated to on-site application. Third, the deployment is flexible and the cost is controllable: the whole system is integrated into a mobile cabinet and can be moved and deployed as needed; with the help of virtualization technology, the engineer station covers two legacy operating system environments with a single physical machine, and the simulation software is developed based on the open-source Python ecosystem, so the construction and maintenance costs are low.

4. CONCLUSION

This paper designs and implements a communication simulation test and verification system based on the Python programming language and virtual machine technology. The system constructs real communication opposite ends with the server-side simulation software for the three protocols MODBUS-TCP, IEC 60870-5-104 and MODBUS-RTU written by Python. On-site application shows that the system can verify item by item the power-on state, communication software operation, link establishment, point-value consistency and telegram byte-level correctness of the communication station, which provides a reference for the construction of off-line verification platforms for I&C equipment of similar nuclear power plants.

The system still has certain limitations: the current protocol simulation covers the three protocols actually used by the communication stations; for communication interfaces of other protocol types such as OPC UA and PROFIBUS, corresponding server-side simulation modules need to be developed separately; In addition, the current verification of the system is mainly functional verification, and the communication stability of long-time operation and the anti-disturbance capability of the link have not yet been quantitatively assessed. Future work will expand the types of protocol simulation and introduce long-period continuous operation tests and fault injection means to further enrich the verification dimensions and improve the completeness of verification conclusions.

REFERENCES

[1] National Technical Committee on Industrial Process Measurement, Control and Automation of Standardization Administration of China. (2008). GB/T 19582.1-2008 Industrial automation

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202611_12(11).0008

networks based on Modbus protocol - Part 1: Modbus application protocol. Standards Press of China.

[2] Modbus Organization. (2006). MODBUS over serial line specification and implementation guide V1.02.

[3] International Electrotechnical Commission. (2006). IEC 60870-5-104:2006 Telecontrol equipment and systems - Part 5-104: Transmission protocols - Network access for IEC 60870-5-101 using standard transport profiles.

[4] National Energy Administration. (2009). DL/T 634.5104-2009 Telecontrol equipment and systems - Part 5-104: Transmission protocols - Network access for IEC 60870-5-101 using standard transport profiles. China Electric Power Press.

[5] Matthes, E. (2016). Python crash course: A hands-on, project-based introduction to programming. Posts & Telecom Press.

[6] VMware Inc. (n.d.). VMware Workstation Pro documentation. https://docs.vmware.com/