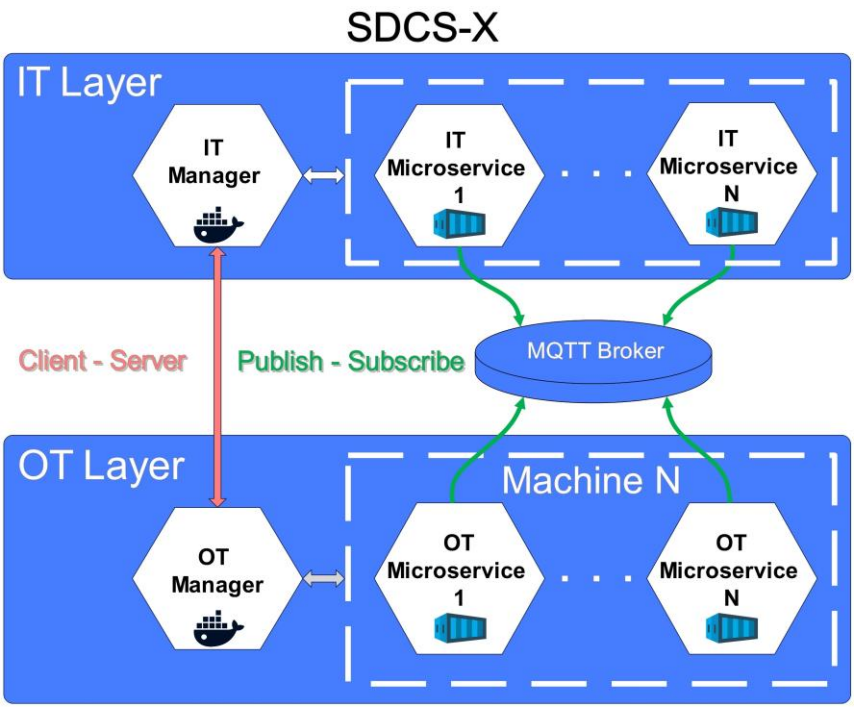
SDCS-X

A framework to enable open and modular Software-Defined Control Systems in Discrete Manufacturing



Author	Shane Ward
Student Number	R00110936
Date Issued	11-May-2025

CONTENT

CONTENT **2**

TABLE OF ABBREVIATIONS **3**

1. PROBLEM STATEMENT **4**

1.1. TRADITIONAL INDUSTRIAL ARCHITECTURES 4
1.2. MODERNISING INDUSTRIAL ARCHITECTURES 5

2. SDCS-X ARCHITECTURE **6**

2.1. WHAT IS SDCS-X? 6
2.2. BENEFITS OF SDCS-X 7
2.3. LIMITATIONS OF SDCS-X 8
2.4. MINIMAL TECHNICAL REQUIREMENTS ..... **ERROR! BOOKMARK NOT DEFINED.**
2.5. CORE CAPABILITIES 8
2.6. EXTENSIBLE CAPABILITIES 8

3. BASE SDCS-X FRAMEWORK **9**

3.1. PREREQUISITES 9
3.2. METHOD 9

4. MIGRATION FROM LEGACY SDCS-X **14**

4.1. COMPONENTS WHICH CAN BE VIRTUALISED 14
4.2. MIGRATION STEPS FOR RETROFIT (BROWNFIELD) PROJECT 14
4.3. MIGRATION STEPS FOR GREENFIELD PROJECT 15

5. CONSIDERATIONS **15**

6. CONCLUSION **15**

TABLE OF ABBREVIATIONS

Term	Description
SDCS	Software-Defined Control System
ICS	Industrial Control System
IT	Information Technology
OT	Operational Technology
MES	Manufacturing Execution Systems
ERP	Enterprise Resource Planning
API	Application Programming Interface
AI	Artificial Intelligence
ML	Machine Learning
IoT	Internet of Things
IIoT	Industrial Internet of Things
OEM	Original Equipment Manufacturer
SI	System Integrator
PLC	Programmable Logic Controller
HMI	Human Machine Interface
DDD	Domain Driven Design
REST	Representational State Transfer
gRPC	Google Remote Procedure Call
MQTT	Message Queuing Telemetry Transport
DDS	Data Distribution Service
OPC	Open Platform Communications
OEE	Overall Equipment Effectiveness

1. PROBLEM STATEMENT

1.1. TRADITIONAL INDUSTRIAL ARCHITECTURES

Industry 4.0 and smart manufacturing initiatives are evolving how manufacturing enterprises utilise technology. Digital Transformation is the process which describes this shift, with manufacturing enterprises aiming to digitally integrate each business layer and utilise data to make informed business decisions, and allow for increased agility to changing market demands.

Prior to Industry 4.0 and smart manufacturing initiatives, manufacturing enterprises relied on the ISA-95 architectural model. This model, shown in Figure 1 below, promotes individual Point-to-Point communication between components at each layer of the automation pyramid. This approach, which does not satisfy the requirements for enabling Digital Transformation, is still widely used in industry today due to industry-specific constraints.

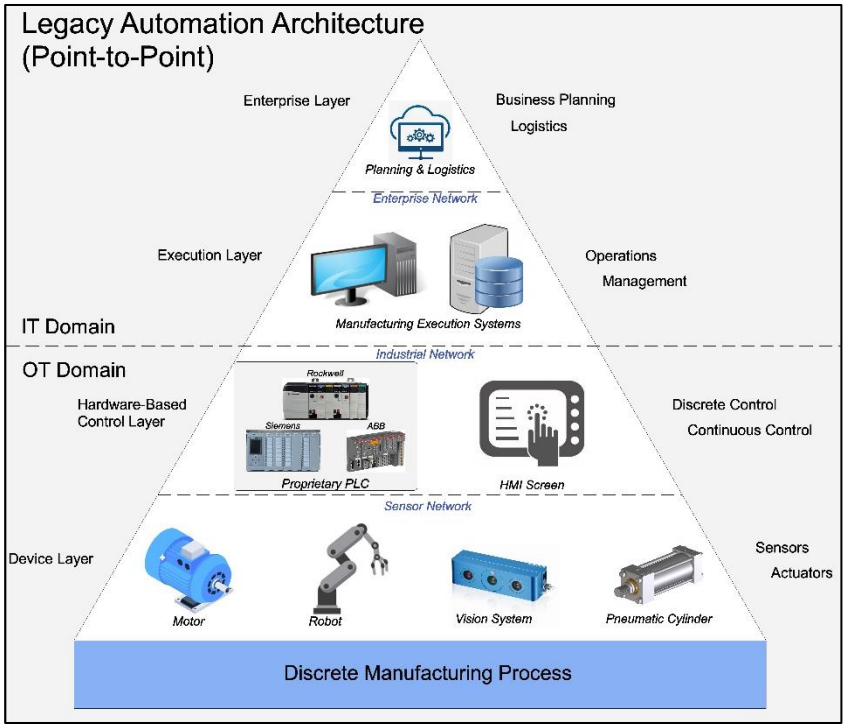


Figure 1 – ISA-95 Architecture – Too Many Dependencies resulting in increased complexity

At the core of all manufacturing enterprises, is the integration of OT systems, which consist of industrial control systems which are designed, programmed and tested to repeatably run machines which operate on the factory floor of manufacturing enterprises. To enable seamless Digital Transformation, existing OT systems must have the ability to be flexible, scalable, and support connectivity with other components within the automation architecture.

OT systems in industrial manufacturing have traditionally relied on legacy control systems with hardware-dependent configurations that offer reliability, but limit flexibility and scalability, primarily due to vendor lock-in. Leading control system providers have long prioritised performance and serviceability, but with the drawbacks of limited openness and scalability for communication with other components at various levels of manufacturing enterprises.

1.2. MODERNISING INDUSTRIAL ARCHITECTURES

Modernising industrial architectures requires shifting from rigid, proprietary systems to agile, software-defined solutions. Traditional automation relies on dedicated hardware and vendor-specific software applications, making it costly, inflexible, and hard to integrate with modern IT systems. Modern OT systems seek to decouple control logic from hardware, enabling cross-platform deployment or virtual environments like Docker. This presents the possibility for manufacturing enterprises to modernise incrementally by retrofitting legacy systems with virtual controllers and digital interfaces—reducing disruption to existing operations.

Modernisation of manufacturing enterprises requires a bottom-up approach, starting with the OT layer to reflect factory floor needs. Traditional hardware control system components such as PLCs and HMIs must be replaced with software-based alternatives such as containerised virtual PLCs (vPLCs), allowing existing control logic to remain, but to allow for additional functionality which is not currently supported such as data collection, connectivity with modern protocols, analytics, and monitoring. This promotes a microservices based approach, utilising protocols such as MQTT, as a single source of truth for real-time data within the manufacturing enterprise.

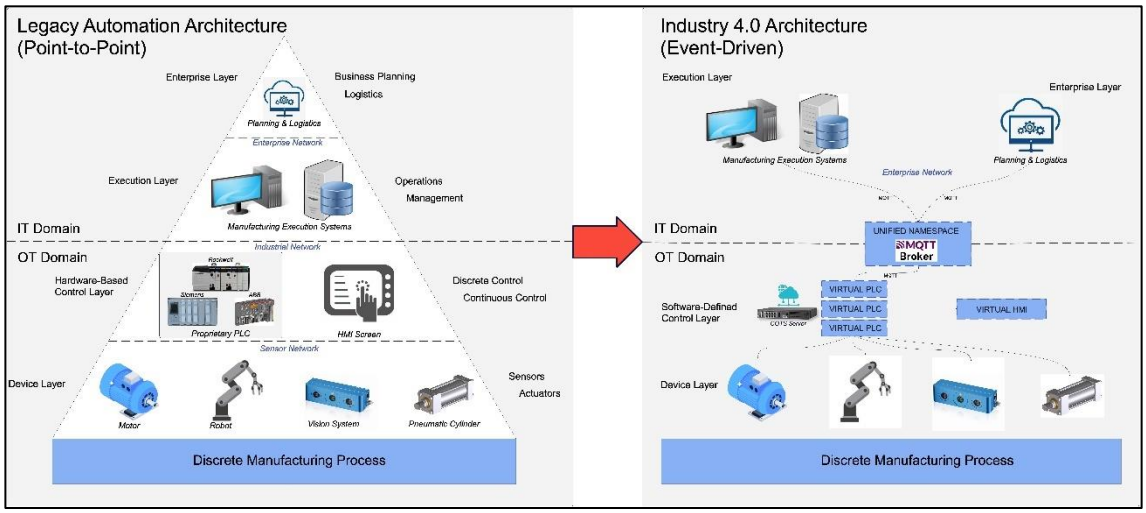


Figure 2 – Moving to an Event-Driven Architecture based on virtual control systems

A primary driver for adopting modernisation is the elimination of vendor lock-in. Traditional control architectures often bind customers to specific hardware and software ecosystems, limiting innovation and increasing long-term costs. Given the vast availability of open-source tooling within the IT domain, being hardware-agnostic and based on open-source or open-standard components within the OT domain, allows manufacturers to choose best-in-class solutions without being tied to a single vendor. This open, composable shift accelerates innovation and enables collaboration across IT and OT domains.

Modernisation enhances scalability and resilience by using containerised microservices and virtual controllers that can scale across edge or central systems. Fault tolerance improves with automated health checks, recovery, and distributed storage. Cloud and IT integration enables secure data flow for analytics and remote diagnostics, laying a foundation for Industry 4.0 built on adaptability, intelligence, and connectivity.

2. SDCS-X ARCHITECTURE

2.1. WHAT IS SDCS-X?

SDCS-X is an extensible framework to enable open and modular Software-Defined Control Systems in Discrete Manufacturing. SDCS-X is an automation architecture which considers all levels of manufacturing enterprises, from the IT to the OT domain. The architectural pattern is based on Docker, which allows for an entire machine to be configured with a single Docker-compose command. Each component, as shown in Figure 3 below, is built as an individual Docker container and executes a specific function whether in the IT or OT Domain. Containers are split into two networks, with secure client server protocols used for management, and publish subscribe protocols used for real-time data transfer to the broker. This allows for the alignment of data through all manufacturing enterprise applications in the architecture.

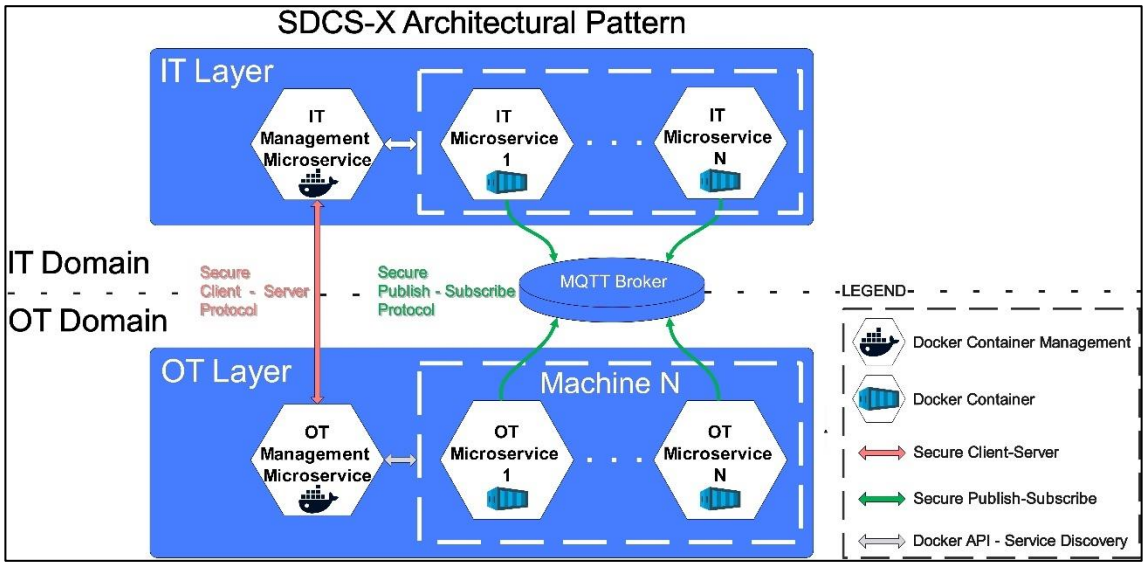


Figure 3 – Supporting real-time data alignment across enterprise applications

SDCS-X allows for the replacement of industrial control system components which utilise proprietary hardware and software by introducing a flexible, containerised architecture. Instead of being tied to specific vendor ecosystems, control logic, interfaces, and services are encapsulated in lightweight, platform-agnostic containers. Within the Operational Technology (OT) domain, these containers are logically grouped into *Machines*, each representing a functional unit of the production process—such as a robot cell, conveyor, or painting booth. Each Machine comprises multiple microservices, such as a virtual PLC (vPLC), HMI (vHMI), historian (vHIS), or vision system (vVIS), all orchestrated and monitored independently.

Multiple Machines are then grouped into a *Production Line*, reflecting the actual layout and flow of a manufacturing process. This structure provides clear modularity and separation of concerns, allowing engineers to deploy, monitor, and upgrade individual Machines without affecting the rest of the system. The Production Line grouping also allows for uniform communication, synchronization, and orchestration of processes across Machines, all coordinated through open protocols like MQTT, OPC UA, or DDS. This organisation mirrors real-world manufacturing hierarchies while introducing the benefits of modern software engineering—scalability, version control, and updates.

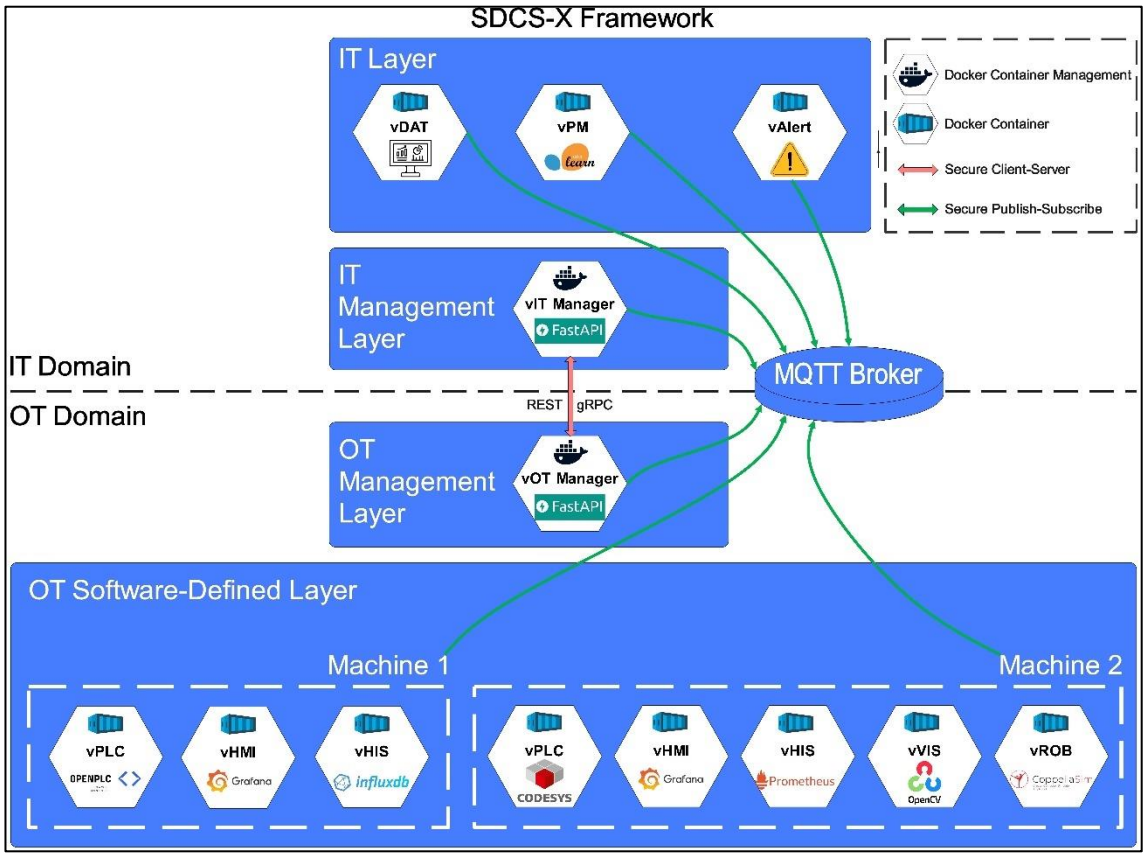


Figure 4 – Base SDCS-X Framework with Machine 1 – Robotic Depalletizer Pre-Configured

Base System Components:

- MQTT Broker: Eclipse Mosquitto
- IT Domain:
  - vIT Manager: FastAPI UI
  - vAlert: Grafana Alert publishing via MQTT
- OT Domain:
  - vPLC: OpenPLC, or CodeSys option
  - vHMI: Grafana
  - vHIS: InfluxDB or Prometheus
  - vVIS: OpenCV
  - vROB: CoppeliaSim or ROS

2.2. BENEFITS OF SDCS-X

- Provides engineers and system integrators with purpose-built tools to support the modernisation of Industrial Control Systems (ICS) and facilitate Digital Transformation initiatives e.g. loading PLC software, automated PLC tag loading.
- Promotes an open, modular, and vendor-agnostic approach to industrial automation, reducing dependence on proprietary systems.
- Scalable across machines, production lines, and multi-site enterprise environments through containerised microservices and orchestration.
- Supports DevOps workflows for continuous integration, deployment, and monitoring of industrial software components.

- Integrates seamlessly with modern, open industrial communication protocols such as MQTT, OPC UA, and DDS, enabling interoperability and future-proofing.

2.3. LIMITATIONS OF SDCS-X

- Requires a cultural and skillset shift towards software-centric practices, including knowledge of containers, networking, and DevOps tooling.
- Initial setup may be complex for teams unfamiliar with container orchestration, protocol mapping, and edge computing infrastructure.
- Integration with legacy systems may require protocol converters or middleware, potentially increasing deployment time and cost.
- Real-time performance in virtualised environments may not always match dedicated hardware in highly time-sensitive applications without careful configuration and tuning.

2.4. CORE CAPABILITIES

- Decouples control logic from hardware, enabling software-defined operation of control systems across heterogeneous environments.
- Manages and discovers IT and OT microservices in a segregated yet unified architecture, supporting domain-specific concerns (e.g., vPLC in OT, logging in IT).
- Enables container lifecycle management, including deployment, health monitoring, fault recovery, and resource optimisation across edge and central systems.
- Provides secure, role-based access control, configuration management, and traceable communication between all components using standard protocols.

2.5. EXTENSIBLE CAPABILITIES

- Allows custom container definitions for specific Machine functions (e.g., quality inspection, digital twin, vision systems, robotics control), extending functionality with minimal disruption.
- Supports plug-in architectures for third-party integration, including analytics platforms, MES/ERP systems, and AI/ML pipelines.
- Facilitates automated data collection and publishing to time-series databases (e.g., InfluxDB), visualisation tools (e.g., Grafana), or alerting systems.
- Enables horizontal scaling through orchestration platforms like Docker Swarm or Kubernetes, and vertical extension through cloud integration and edge-device deployment.

3. BASE SDCS-X FRAMEWORK

3.1. PREREQUISITES

Recommended System Requirements:

OS:	Linux (For now)
RAM:	8GB
Storage:	20GB

validsight@validsight-Z590-UD-AC: ~

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
21f45afde73c	vhmi	0.23%	69.49MiB / 62.66GiB	0.11%	196kB / 400kB	991kB / 725kB	31
e70c8a4bfb6b	vplc	3.16%	49.79MiB / 62.66GiB	0.08%	17.8MB / 41.9MB	807kB / 2.88MB	13
45076ba7292d	vhis	1.75%	171.6MiB / 62.66GiB	0.27%	5.36MB / 3.46MB	24.6kB / 219MB	32
3772d2d50349	valert	0.27%	30.79MiB / 62.66GiB	0.05%	48.7kB / 50.3kB	0B / 168kB	2
3f8d509f275d	vot_manager	0.72%	57.25MiB / 62.66GiB	0.09%	23MB / 13.9MB	123kB / 934kB	3
8d5932fa0f19	vit_manager	0.62%	55.61MiB / 62.66GiB	0.09%	22.7MB / 13.6MB	32.8kB / 889kB	2
d2e72dc06c99	mqtt_broker	0.37%	6.434MiB / 62.66GiB	0.01%	46.1MB / 54.6MB	5.8MB / 0B	1
224f8695e535	portainer	0.00%	70.46MiB / 62.66GiB	0.11%	1.47MB / 18.4MB	47.3MB / 9.86MB	22

Figure 5 – Snapshot of Running System

Software Requirements:

Docker
Portainer CE

3.2. METHOD

1. Clone SDCS-X Framework github repository:
   - https://github.com/shanewardmsc/SDCS-Framework/tree/main/SDCS-X

2. Navigate to docker compose directory:
   - cd SDCS-X/Docker_Compose

3. Option 1: Run the Base SDCS-X Framework using Docker-compose:
   - Open Terminal
   - Type the command: *sudo docker-compose up –build*

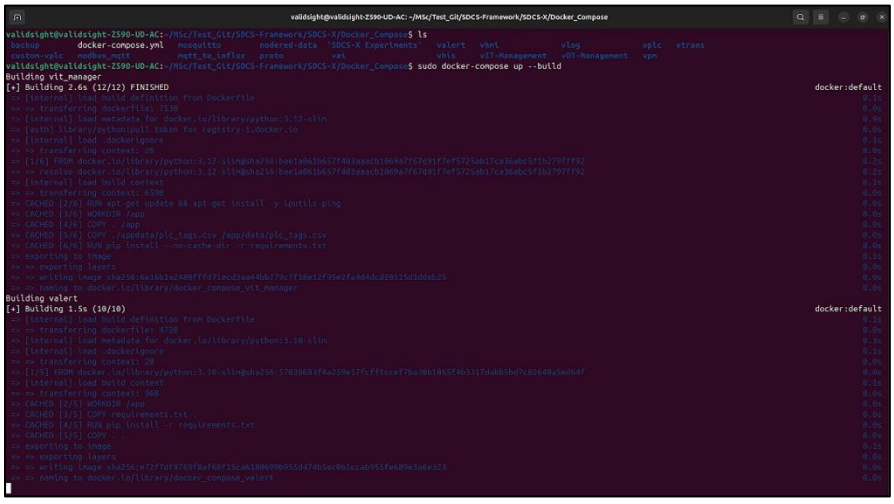


Figure 6 – Docker Compose Build Underway

- Ensure Portainer container is running
- Navigate to Portainer via web browser
- Login using credentials
- Select Environment
- View SDCS-X Containers by navigating the Container tab

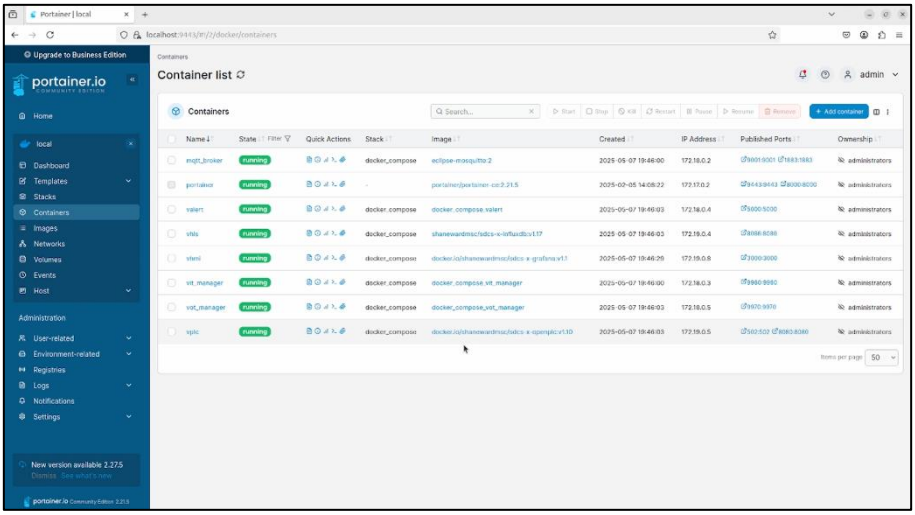


Figure 7 – SDCS-X Containers running on Portainer

4. Option 2: Run the Base SDCS-X Framework using Portainer Stack:
   - Ensure Portainer container is running
   - Navigate to Portainer via web browser
   - Login using credentials
   - Select environment
   - Select Stacks tab
   - Create new stack called sdcs_x or similar
   - Select Upload build method

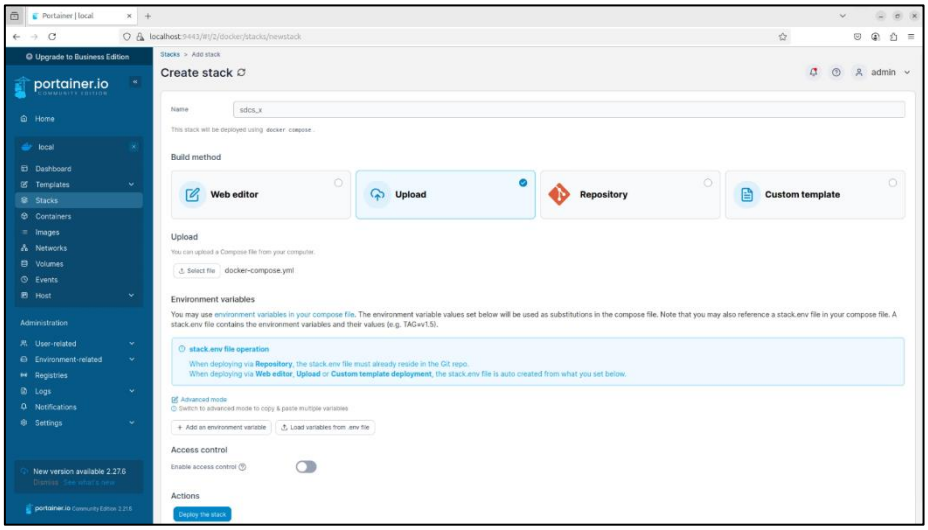


Figure 8 – Creating SDCS-X Stack using Portainer

- Upload Docker-compose file from relevant directory
- Run defined sdcs_x Stack
- View SDCS-X Containers by navigating the Stack tab

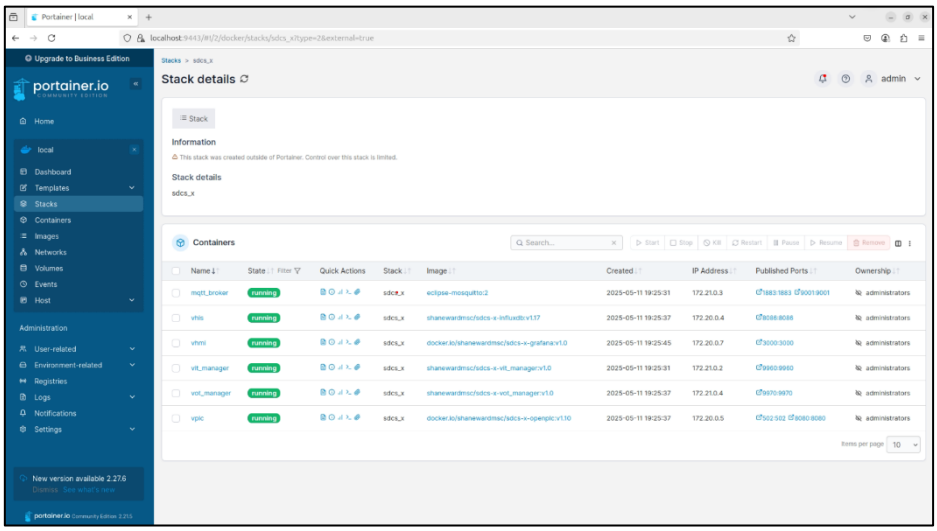


Figure 9 – Viewing SDCS-X Stack using Portainer

3.3. vOT Manager

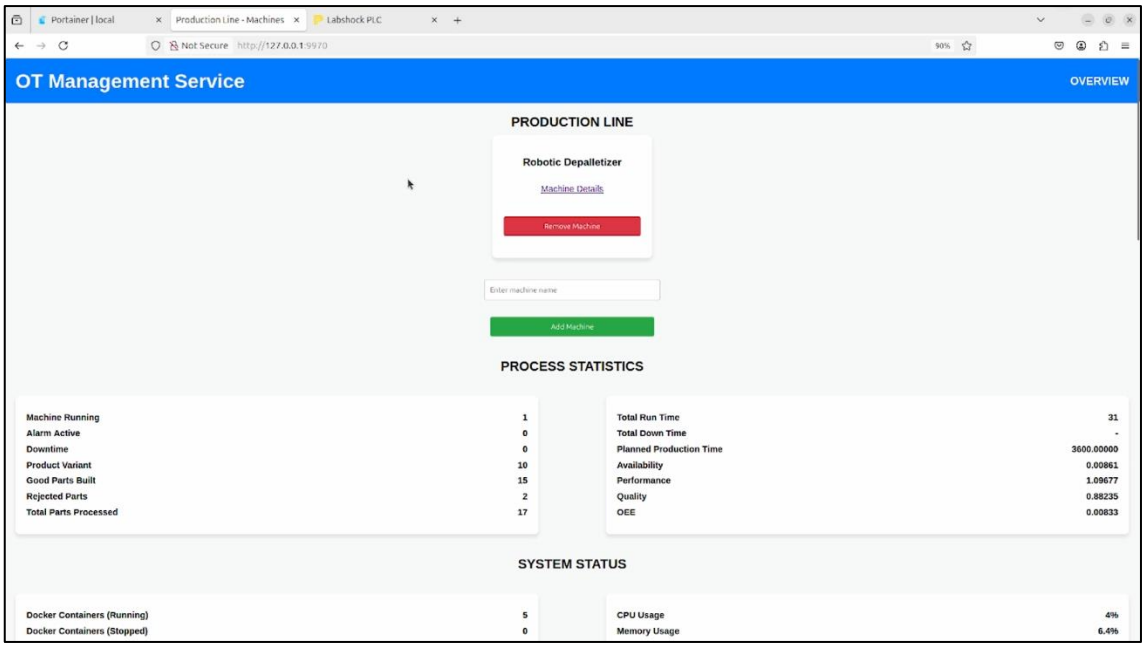


Figure 10 – vOT Manager for Management of OT Containers

3.4. vPLC

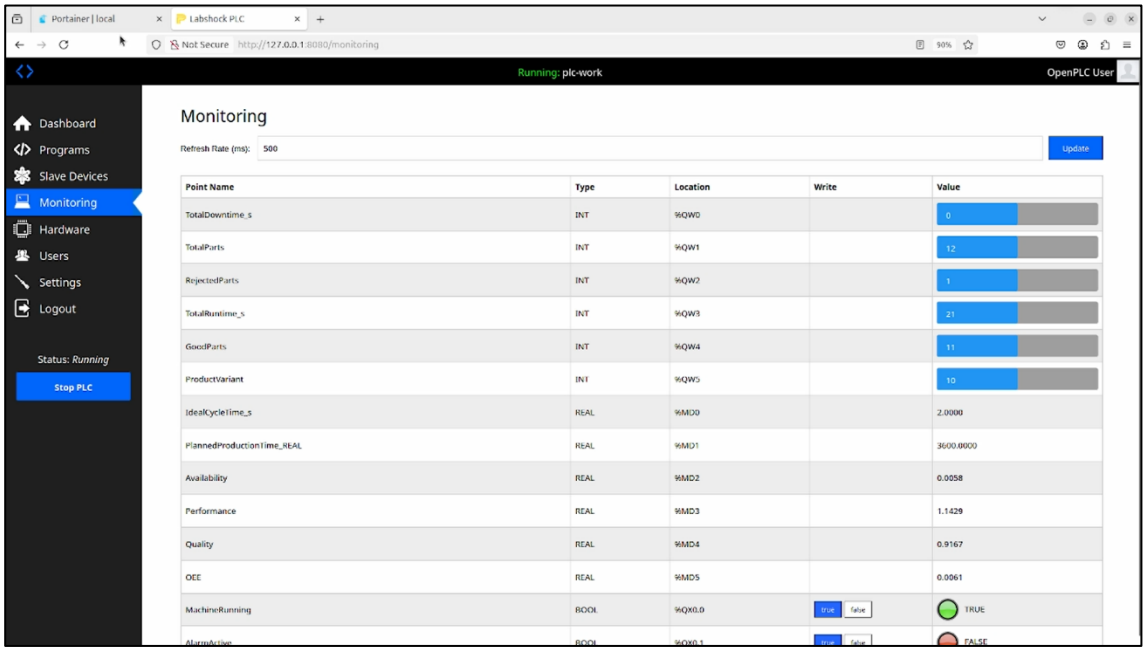


Figure 11 – vPLC for Execution of Machine Logic

3.5. vHMI

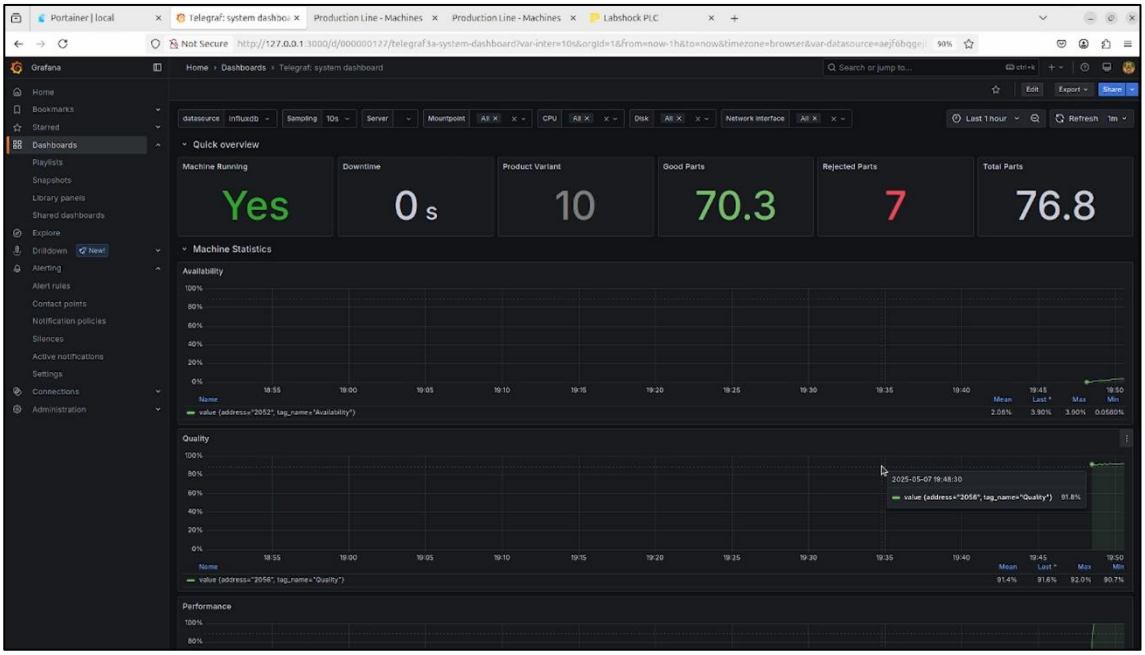


Figure 12 – vHMI for User Control of Machine via Graphical Display

3.6. vHIS

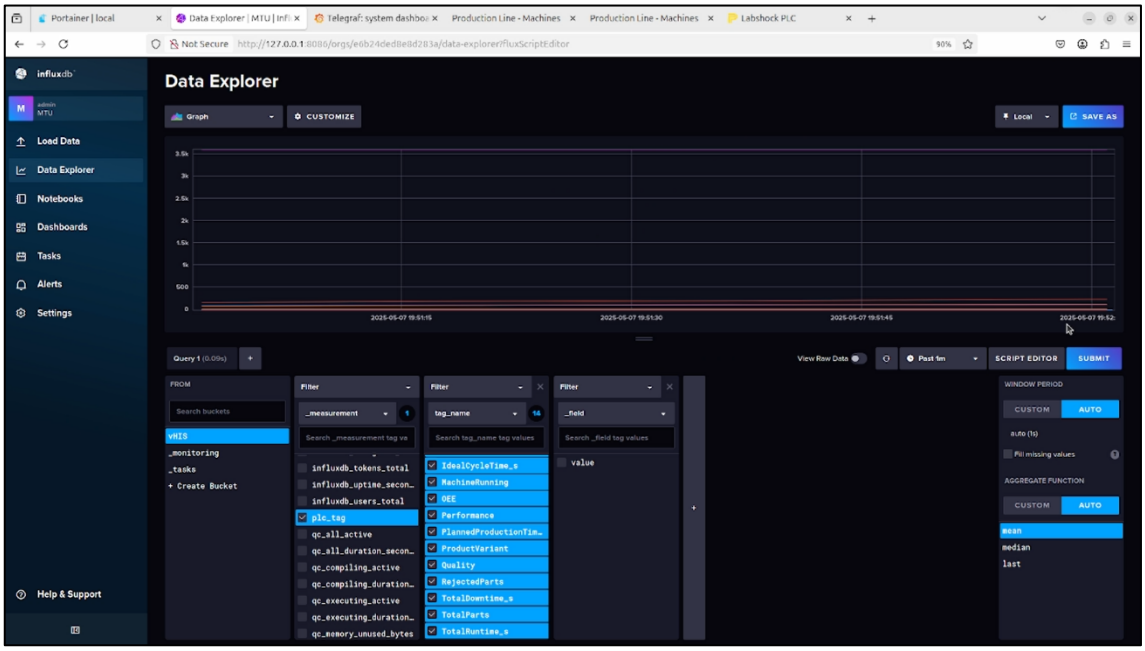


Figure 13 – vHIS for Storage of Time Series data i.e. PLC Tags

3.7. vIT MANAGER

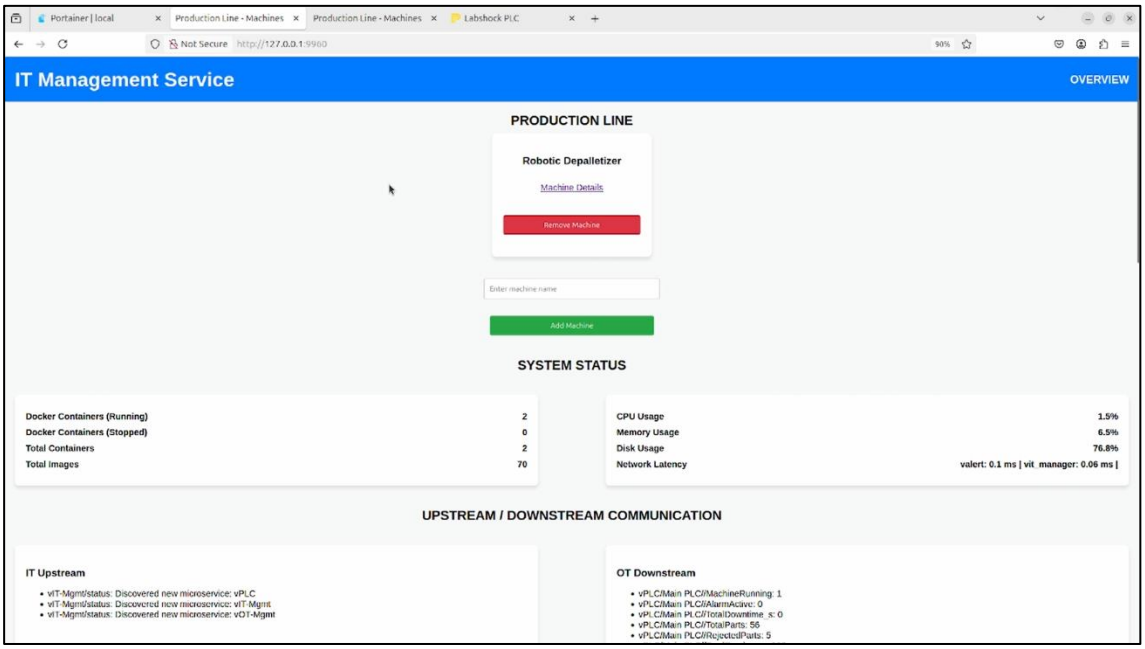


Figure 14 – vITi Manager for Management of IT Containers

4. MIGRATION FROM LEGACY SDCS-X

4.1. COMPONENTS WHICH CAN BE VIRTUALISED

Level 1 Components:

- Level 1 Components are controllers which perform deterministic logic for specific devices in Level 0 (proprietary).
- Existing control components such as PLCs, HMIs, Robots, Vision, Remote IO Adapters, (shown in Figure below highlighted in green) can be virtualized as they only communicate to downstream using industrial protocols, and execute standardized IEC-61131-3 logic.
- The above functionality can be achieved using non-proprietary hardware and software and using open automation protocols – therefore reducing cost, and allowing for more connected smart automation systems.

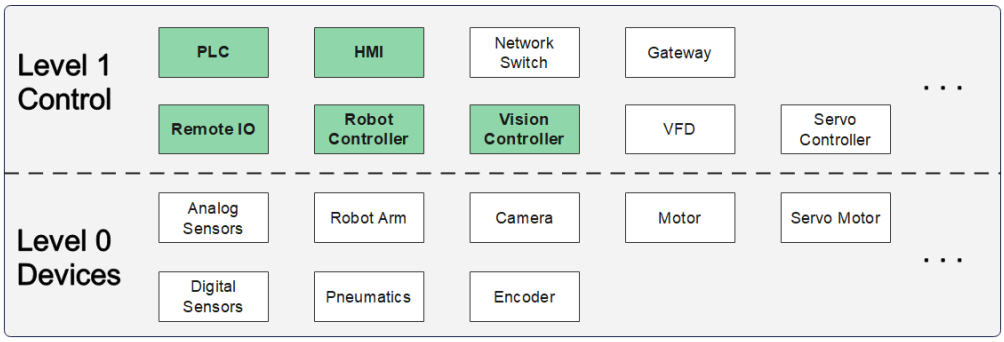


Figure 10 – Highlighted components which can be virtualised

4.2. MIGRATION STEPS FOR RETROFIT (BROWNFIELD) PROJECT

1. Convert / Update PLC Program
    1.1 Remove vendor lock-in
    1.2 Leverage existing PLC Program
    1.3 Select IDE
2. Select Hardware for Running vPLC
    2.1. Choose the devices for running the vPLC – SDCS-X recommends to use a GPU enabled IPC to support multiple microservices on a single device.
    2.2. Decouple I/O from legacy PLC – Use Remote I/O module for rewiring if required.
    2.3. Select Network hardware.
3. Visualisation
    3.1. Migrate HMI screens from proprietary HMI Panels to open-source powerful visualization tools such as FastAPI, Grafana, Ignition which support modern communication protocols.
4. Select Third Party Applications
    4.1 Choose vPLC vendor which supports modern pub-sub protocols to enable bi-directional communication.
5. Enable Remote, Smart Orchestration
    5.1. Enable centralized software management and deployment
    5.2. Enable plant-wide vPLC (or other microservice) operations for monitoring, troubleshooting, and any additional desirable functionality in a modular way.

4.3. MIGRATION STEPS FOR GREENFIELD PROJECT

1. Select Hardware for Running vPLC
    1.1 Choose the devices for running the vPLC – SDCS-X recommends to use a GPU enabled IPC to support multiple microservices on a single device.
    1.2 Use Remote I/O module to decouple sensor wiring.
    1.3 Select Network hardware.
2. Visualisation
    2.1. Utilise open-source powerful visualization tools such as FastAPI, Grafana, Ignition which support modern communication protocols.
    2.2. Create local and remote dashboards.
3. Select Third Party Applications
    3.1. Choose vPLC vendor which supports modern pub-sub protocols to enable bi-directional communication.
4. Enable Remote, Secure, Smart Orchestration
    4.1 Enable centralized software management and deployment.
    4.2 Enable zero-trust security with role-based access control or certificate authentication.
    4.3 Enable plant-wide vPLC (or other microservice) operations for monitoring, troubleshooting, and any additional desirable functionality in a modular way.

5. CONSIDERATIONS

- Ensure to adhere to naming convention when extending the SDCS-X framework.
  The framework operates so that each container is isolated per control system function.
- Ensure network segregation and appropriate user authentication principles are adhered to when extending the SDCS-X framework.
  Security is an essential component of the framework, however this must be maintained.
- Ensure to include publish appropriate messages when extending the SDCS-X framework to be able to benefit from added functionality.

6. CONCLUSION

SDCS-X provides a modern, scalable, and modular foundation for software-defined control in discrete manufacturing. It accelerates digital transformation by enabling flexible orchestration of control logic, data acquisition, and visualization across legacy and modern equipment using containerized microservices.