AOLE Project Report

TOPIC: REMOTE CONTROL VIRTUAL MODEL OF AN AUTOMATION-RELATED SETTING THOUGH OPC UA WITH CLOUD DATA VISUALIZATION

Minh Duc Pham, AEE - Controls, Robotics and Autonomous Systems
Student number: 796987

Advisor: Udayanto Dwi Atmojo
Supervisor: Valeriy Vyatkin

06.01.2021

Aalto University

AFOF - Aalto Factory of the Future

1 Information page

Abstract

In this document, the report on the AOLE - Aalto Factory of the Future in the Cloud, is presented. The background, methods, critical findings are described based on the evidences gathered during the project. In the end, the document will be concluded with a proposal for future development.

AOLE project management

The project is called "AOLE - Aalto Factory of the Future in the Cloud", initiated and supervised by Utdayanto Dwi Atmojo and conducted by Minh Duc Pham. The project was executed entirely remotely with weekly checkpoints over Microsoft Teams meetings and text chat. The project lasted for three months, from September 2020 to November 2020.

Project Log

Period	Date	Author	Description
0.1	09-2020	Minh Duc Pham	Problem statement, Tools setup, Simulation developing
0.2	10-2020	Minh Duc Pham	APIs, OPC UA and MQTT, Hypothesis introducing
0.3	11-2020	Minh Duc Pham	Architecture forming, Hypothesis analysis, Conclusion

Table 1: Project diary

Contents

2 Introduction

The scope of this project is to create a virtual representation of an automated production layout on Visual Components where users can interact, simulate and make modifications through a set of APIs. These APIs will be created based on the granular level of control, ranging from individual control to sequence control. The intercommunication between Visual Components and the middlewares involved is feasible via industrial communication protocol (OPC UA) and IoT messaging protocol (MQTT). The corresponding APIs can be called from OPC UA Server and the desired data of the simulation and layout can be visualized on OPC UA Client and on Cloud (ThingSpeak).

2.1 Problem description

Prior to COVID-19 introducing major changes to studying, students were able to attend lectures and laboratories sessions working face to face or they can reserve study spaces via the Aalto Space - mobile app [**AaltoSpace**] to access licensed software and necessary tools. However with current situations, such practice is prohibited in order to maintain everyone's well-being. As a result, remote laboratories sessions are in need, thus poses new challenges for the University personnel. This project offers an intuitive online means for students to interact with a model automation layout, learning how to use APIs to create meaningful control processes in Visual Components. On top of that, they will not only have hands-on experience using industrial automation protocol, but also will be able to understand the information systems in industry. Last but not least, a nice data visualizations in real-time will give them a quick glance into the trendy Cloud technology. The following architecture is suggested in order to achieve all the above points.

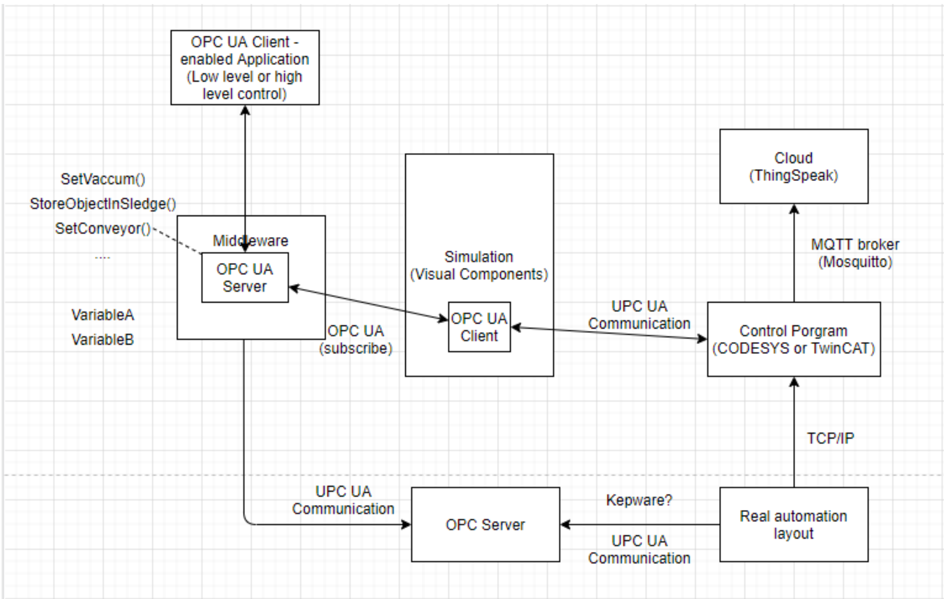

Figure 1: Architectural proposal

3 Methods

After having the architecture finalized, the implementation of different parts has to be planned. Overall, the architecture can be broken down into four main elements.The first central element, which will be the core of everything. The second element would be the OPC Servers and OPC Clients, which will be the backbone of the whole information systems throughout this project. The third element is the Beckhoff TwinCAT environment where the logic for data retrieval from Visual Components to Cloud lies. Last but not least, the desired data from Visual Components will be published on Cloud, the fourth element, through the use of MQTT and Mosquitto.

3.1 Visual Components (VC)

Visual Components' 3D manufacturing simulation solutions are trusted by hundreds of organizations worldwide to support critical planning and decision-making processes. The high-performance 3D simulation engine and open API enable students to build or customize a model simulation solution into their own. A lot of common process components, machines, robots, etc. are available on VC or we can design and import our own designs into the environment. This is a great learning platform for students and bring with it so much potential for future development. Its interoperability with other programs through standard OPC UA support enables smooth communication. Last but not least, Aalto has purchased a license for it, which is accessible through VMWare, thus making it suitable for the scope of this thesis.

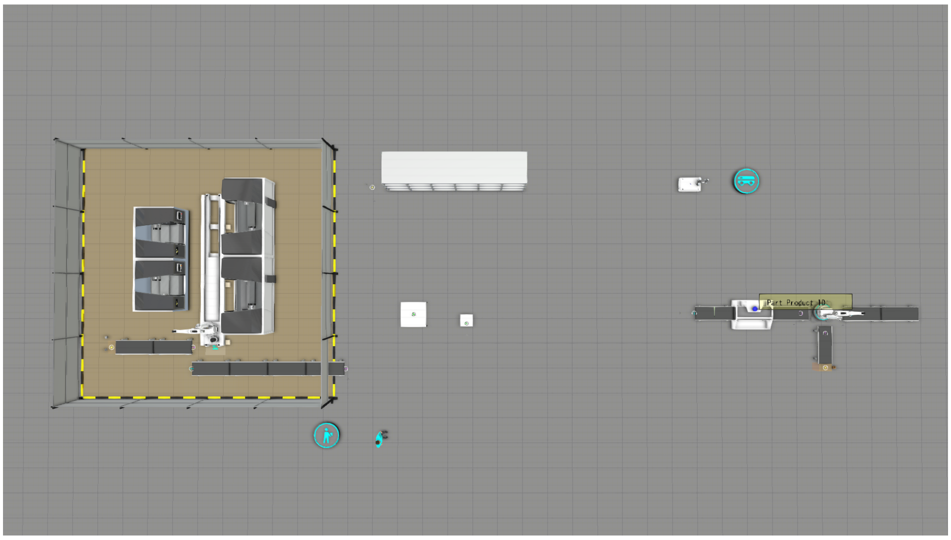

Figure 2: Suggested VC layout

The layout above involves all the common units every manufacturing factories with production line, assembly line and quality control line. The present of heavy-duty machines, conveyor belts, storage self, robot arms, human workers, AGV (Mir100 with UR3e robot arm on top just like the model in AFoF) and camera makes the process more realistic and relatable.

3.2 OPC UA Server and OPC UA CLient

OPC UA (OPC Unified Architecture) is the interoperability standard for industrial automation. OPC Server, which makes its data available in the network, can be addressed via protocols such as TCP/IP and web protocols for communication. The OPC Server can be connected to the OPC Client and read out the data provided by the Server. Since the OPC Servers implement the predefined interfaces of the OPC standard, each client can access any OPC Server and exchange data with the server in the same way. Here in this thesis, almost all of the communication links are made possible through OPC UA. The figure below shows a validated interconnection scenario.

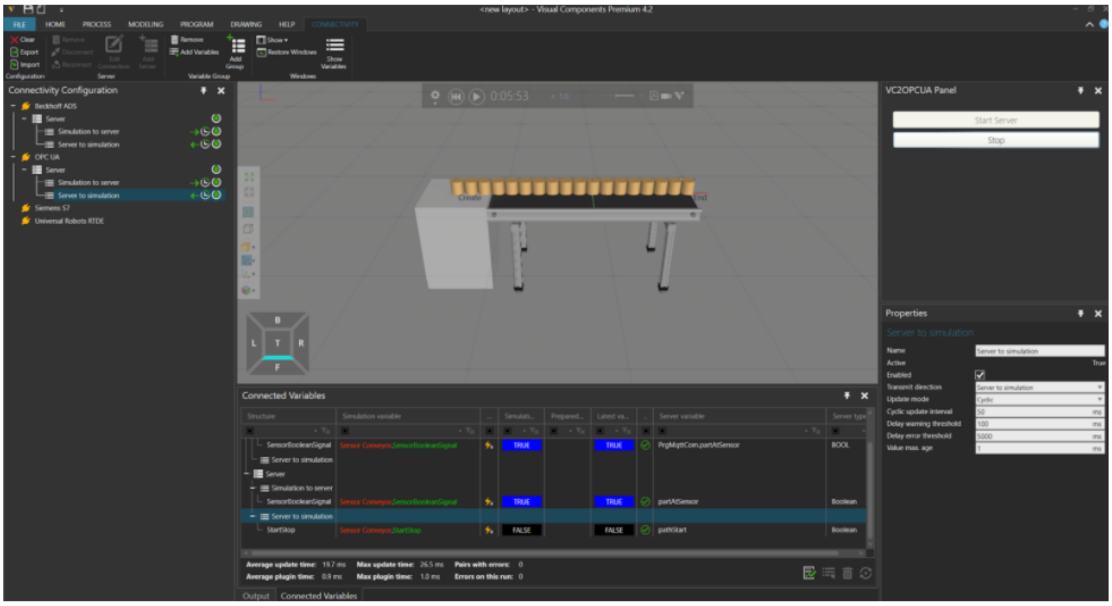

Figure 3: Example on VC interconnections

In this figure, VC has successfully connected to two different OPC servers hosted on TwinCAT and CODESYS respectively. On top of that, VC is hosting its own OPC UA Server through the help of a custom-made add-on. According to VC, the communication to an OPC UA Server will always be enabled provided that it was created corresponding to the OPC UA standard certificate specification.

3.3 Beckhoff TwinCAT

The Beckhoff TwinCAT software system turns almost any compatible PC into a real-time controller with a multi-PLC system, NC axis control, programming environment and operating station. TwinCAT replaces conventional PLC and NC/CNC controllers as well as operating devices. Here in this thesis, TwinCAT will serve as a helping-hand to VC logic control, as well as a bridge between VC to MQTT and ultimately to Cloud. In the Figure 3 scenario, a sensor embedded on the conveyor belt is linked to a counter in TwinCAT, thus storing data in TwinCAT counter every time a piece enter the conveyor. It also shows here its control capability through controlling the start and stop of the conveyor belt.

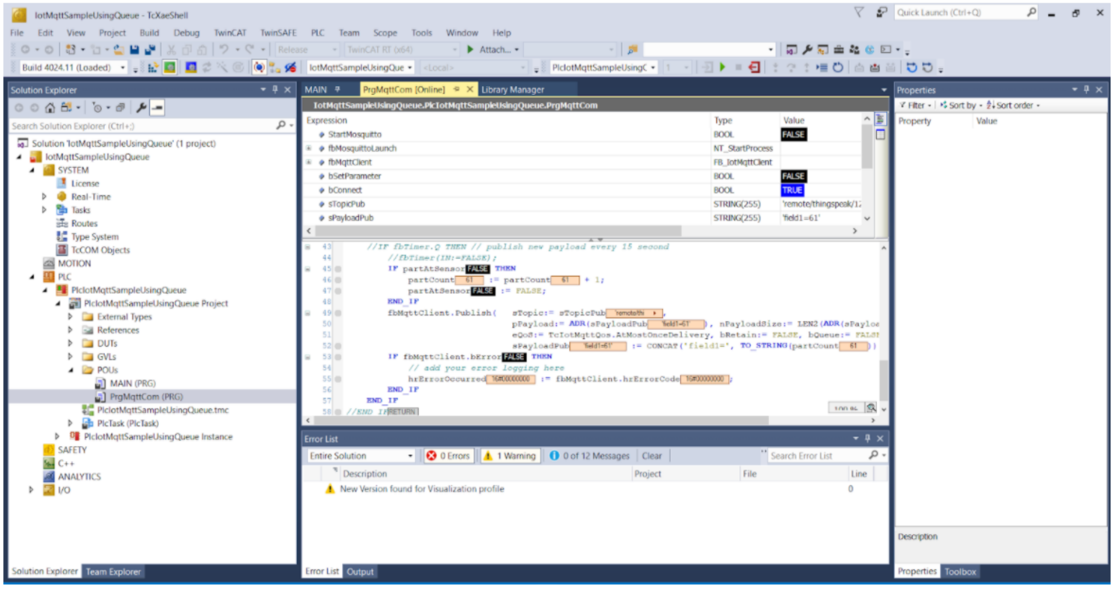

Figure 4: Example on TwinCAT role

Here in figure 4, this piece of ST code establishes MQTT connection with ThingSpeak through Mosquitto message broker and controls the conveyor belt on VC environment.

3.4 MQTT, Mosquitto and Cloud (Thingspeak)

MQTT (MQ Telemetry Transport or Message Queuing Telemetry Transport) is a major IoT protocol in public cloud systems. It runs on TCP/IP using publish-subcribe model that uses a message broker for handling/serving the subscribers and publishers. Industry 4.0 uses MQTT and OPC UA as communication to provide service oriented architecture to applications. Mosquitto is a message broker that is used as a bridge from MQTT to ThingSpeak. ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by the devices to ThingSpeak.Here in this thesis, ThingSpeak will be utilized as a Cloud platform to visualize desired data taken from the simulation on VC.

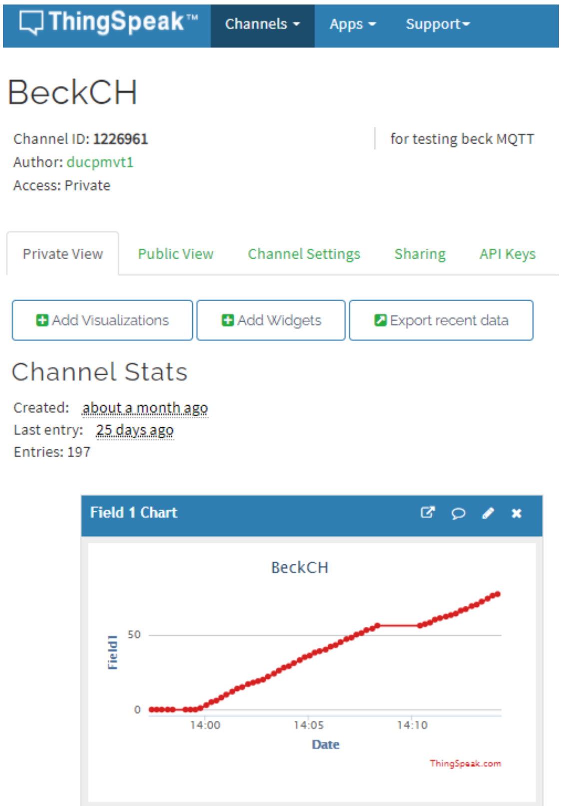

Figure 5: Example on ThingSpeak data visualization

The figure above visualizes the counter values on the parts passing the conveyor belt in VC simulation as shown in Figure 3.

4 Analysis

From the AOLE project we have validate the followings, which proves the proposed architecture feasible:

1. Visual Components can connect to at least two OPC UA Servers while being hosted as an OPC Server itself
2. The interconnection between Visual Components, TwinCAT and Cloud has successfully been established
3. OPC UA Server can call on OPC Clients APIs through Method Node
4. Visual Components supports APIs creation and customization in Python or .NET
5. A wide range of OPC UA vendors provide C++ SDK, Java SDK or .NET SDK for custom-made OPC UA development

5 Conclusions

Based on the findings above, I would like to believe that the proposed architecture as shown in Figure 1 is possible to achieve. As continuation of this idea, I would need to further develop the layout in Visual Components environment, a set of APIs corresponding to the granular control of the layout, an OPC UA Server which serves as a middleware for calling APIs on Visual Components and most importantly, the interconnection between all the elements in the overall architecture. As a mean to improve consistency of this idea, I would like to attempt developing the APIs and the OPC UA Server using the common language in .NET programming.