IoT Integration in the Manufacturing Environment Towards Industry 4.0 Applications

Christos Alexakos, Andreas Komninos, Christos Anagnostopoulos,
George Kalogeras and Athanasios Kalogeras
Industrial Systems Institute
ATHENA R.C.
Patras, Greece
Email: alexakos@isi.gr, komninos@isi.gr, anagnostopoulos@isi.gr,
gkalogeras@isi.gr, kalogeras@isi.gr

***Abstract*—The manufacturing environment undergoes a disruptive evolution due to the Fourth Industrial Revolution driven by the Industrial Internet of Things and Cyber Physical Systems technologies. This evolution is applicable to further sectors comprising similar requirements, involving large numbers of devices that need to interoperate, exchange their data and be controlled. Integration at the manufacturing environment remains a challenge taking into account the diversity of equipment / devices, the existence of legacy systems, and the need to integrate IoT devices participating in the production paradigm. This paper presents an AutomationML based approach for this integration, modeling the industrial manufacturing environment, and enabling its emulation through a 3D Virtual Environment.**

***Index Terms*—Manufacturing Environment, AutomationML**

I. Introduction

The concept of utilizing computing and networking equipment to enable monitoring and control of different types of devices has been around and evolving for several decades. This has given birth to the term Internet of Things (IoT) representing a paradigm in which a wide array of everyday objects or "things", often equipped with ubiquitous intelligence, are interconnected via networks [1]. One of the first sectors that exploited IoT possible benefits through the interconnection of a large number of devices is the industrial/manufacturing sector. Such is the impact of IoT technology integration in manufacturing, that has led to disruptive advancements deemed as the Fourth Industrial revolution. IoT applications were rapidly integrated into industrial products. Large and complex CyberPhysical Systems (CPS) over the Industrial Internet of Things (IIoT) have increased the Fourth Industrial Revolution impact in diverse areas, including further to manufacturing, among others critical infrastructure protection, smart buildings and cities, and health. The manufacturing environment has undergone a profound transformation through innovative and highly agile products and services, that can become partially independent, responsive and interactive, track their activity in real-time and optimize whole value chain via relevant status information throughout their lifecycle [2].

A major challenge for the application of the Fourth Industrial Revolution digitization concept, and more specifically Industry 4.0, in manufacturing is the integration of all legacy systems with new systems and networks, so as to efficiently collaborate to the end of providing flexible manufacturing services. Legacy control systems, IIoT and IoT devices interconnected via different type of networks (Wifi, Modbus, Ethernet, etc.) must expose their functionalities to other systems for utilization in a context of advanced manufacturing paradigms such as collaborative manufacturing. Towards this end, OPC UA serves as an open standard for communication of manufacturing control devices with third systems. Even in the case that systems residing at the lower layers of the classical automation pyramid are interoperable, a unified model is needed defining the overall integrated manufacturing environment including assets, functionalities/operations and manufacturing processes. AutomationML fulfills most of these requirements and allows modeling of the manufacturing environment in a wider perspective defining assets and relationships.

This paper presents an approach for integration of legacy industrial equipment, IoT and IIoT Devices that need to collaborate in the context of the manufacturing production process. The approach uses AutomationML as engineering standard to model shop floor layer. The generated model is able to be realised and deployed in real production environment, as it permits automatic configuration of core functional components of the architecture. Moreover, the proposed architectural scheme makes possible automatic emulation of the designed model in a 3D Virtual Manufacturing Environment, driving towards a Digital Twin approach of virtual representation of manufacturing "things" operation and living throughout their lifecycle [3]. This Virtual environment can be used to depict the execution of real production processes.

The paper is structured as follows. Section II presents briefly major technologies and methodologies for Industry 4.0 concept application. Section III describes manufacturing environment

We acknowledge support of this work by the project I3T - Innovative Application of Industrial Internet of Things (IIoT) in Smart Environments (MIS 5002434) which is implemented under the "Action for the Strategic Development on the Research and Technological Sector", funded by the Operational Programme "Competitiveness, Entrepreneurship and Innovation" (NSRF 2014-2020) and co-financed by Greece and the European Union (European Regional Development Fund).

copyright2020 IEEE

modeling via AutomationML. Section IV presents proposed system architecture and its operation under configuration and execution modes. For proposed approach evaluation, a hybrid environment of real and virtual components is used and outcomes are presented in Section V. Finally, conclusions and future work are presented in Section VI.

II. Towards Industry 4.0

CPS over IIoT have spurred the Fourth Industrial Revolution, leading major industrial players to develop initiatives that promote integration of novel technological concepts into highly intelligent manufacturing processes. Two major such initiatives are Industry 4.0 and the Industrial Internet Consortium (IIC), originating from Germany and USA respectively. A major goal of both is to offer architectural models that serve as standards for application developers. The Industry 4.0 Platform proposed Reference Architectural Model Industry 4.0 (RAMI 4.0) [4]. IIC proposed the Industrial Internet Reference Architecture (IIRA) [5], a more generic approach meant to cover a wider range of IIoT applications.

IoT and cloud computing are cost-effective and flexible solutions. IoT devices like smart sensors, RFID tags, etc. enable immediate collection of data to monitor health of products and equipment in real time [6], [7]. Finally, cloud computing provides a cost effective model for data storage and processing, through a highly flexible support that can grow or shrink dynamically minimizing local management costs.

It is apparent that digitization is one of the key differentiators enabling companies to be competitive in the future, through integrating different technologies, data, applications, devices and resources in general. To facilitate this cause, industrial platform development as part of reference architectures is essential, so that data from various sources implemented by different technologies via well-defined architectures are made available for use by various applications. A platform selection decision is not an easy task and many factors have to be considered. Some criteria may be platform scalability capabilities, whether it is open-source or not, integrability with other tools and platforms, security mechanisms implemented, cost, degree of maturity, and real-time interaction (latency control). A comparison of different digital platforms is provided in [8].

III. Manufacturing Environment Modelling

Manufacturing involves a range of systems related to various operational aspects of the product lifecycle. Diverse software systems relating to product lifecycle management, supply chain management, enterprise resource planning and manufacturing execution systems are often installed as standalone components, facing significant integration challenges, due to lack of a common standard for data exchange [9]. To address this issue, AutomationML standard emerged as part of an industry-led effort to develop and adopt an open standard for data exchange, applied to whole manufacturing lifecycle. Further to its use as a common standard to improve discrete system interoperability in manufacturing, AutomationML has been used as a design tool to develop systems from the ground up [10]–[12] and to enable a range of developments in Industry 4.0, aiming at total digitization of the manufacturing process, including concepts such as the Digital Twin [13], [14].

AutomationML is an XML-based data format, encapsulating use of various other industry-relevant data formats to suit specific purposes. Summarizing AutomationML functionality and design-enabling features, engineers are able to model a manufacturing environment at any level of abstraction, from high-level concepts such as a manufacturing robot, down to individual nuts and bolts that make up any device, depending on desired functionality. AutomationML allows engineers to define individual manufacturing component topology, through properties and relations in hierarchical structures of objects, using the CAEX standard (IEC 62424). Object geometry and kinematics can be specified for each object in its AutomationML entry, using a reference to COLLADA format files. System logic can be implemented using PLCOpen XML.

At the design level, the bulk of the engineering process rests with decisions that need to be taken to model the topology and object hierarchies. AutomationML defines four basic concepts in which design decisions are made. The *Instance Hierarchy* defines the topology of shop floor, defining abstract entities (e.g. a ”milling station”) which can be populated with instances of distinct objects or object instance hierarchies (e.g. ”conveyor belt”, which in turn may include ”belt motor” and ”photoresistor sensor”). Each object is defined by a *System Unit Class*, which describes collections of vendor-specific equipment. Multiple such classes can serve one or more role in the system. Roles are defined using *Role Class* elements, providing semantics to the components of each System Unit Class. Finally, the *Interface Class* elements define commonalities that are are shared between Role or System Unit classes. As can be seen, AutomationML borrows heavily from the concept of object-orientation, allowing the definition of complex class types using nested hierarchies, inheritance, abstraction and encapsulation. This flexibility is not often matched by other components critical to the operation of a smart manufacturing facility, which use less flexible abstraction and modelling concepts. Therefore, AutomationML topologies and relations cannot be readily used without translation and mapping into the particular conceptual constraints of subsystems.

IV. Proposed Architecture

A. General System Architecture

The proposed system architecture is designed for supporting engineers to model the resources, industrial devices or common IoT devices, at the manufacturing shop floor layer along with their operations. The final scope is the automatic configuration of all the monitoring and response infrastructure in order to be ready for deployment in a manufacturing environment. In order to cover the engineering requirements the system supports two types of runtime environments: a) a real shop floor with machinery, sensors and actuators (including humans), and b) a virtual manufacturing environment based on Unity 3D engine. The basic idea behind this decision is the

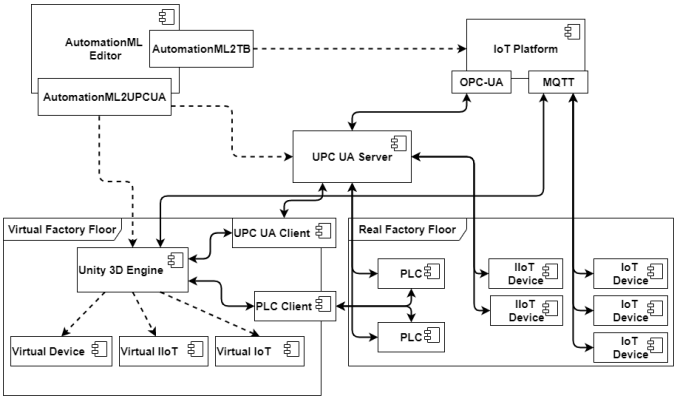


Fig. 1. Proposed System Architecture

expansion of the utilization of the proposed system in more cases than the classical manufacturing automation. Thus, it can operate in three modes, each one with different purposes:

- **Real Factory Floor Execution:** In this mode only the system is used for the integration of all the resources in the shop floor under one centralised entity able to collect data and manage the manufacturing processes.
- **Virtual Factory Floor Emulation** This mode is generally used for purposes of demonstration or simulation. The engineers will be able to test new manufacturing processes or equipment.
- **Hybrid Operation** This mode includes both the operation of a real and virtual environment. The virtual environment can be used as an extension of the real environment, usually for demonstration. Also, the virtual environment can be a digital copy of the real one, putting the basis for a digital twins infrastructure, where engineers can monitor the execution in the real manufacturing environment in a 3D presentation.

Fig. 1 depicts the overall system architecture with its major operating components. Moreover, the interoperation between these components is presented in arrows. Arrows with dashed line show the data exchange during the configuration phase and arrows with straight line represent the communication during the execution phase. Each component is responsible for providing descriptive services inside the system.

IoT Platform is the core component of the system. It contains the registry of all the equipment and production related data. It collects data from both Industrial components and IoT devices, while also it is responsible for exchanging data and commands between the systems in the upper layers to the control systems in the production floor. For the interoperation with industrial equipment, an OPC UA interface is used. A second communication interface based on the MQTT protocol is used for communication with the IoT devices. In the proposed approach the ThingsBoard open source platform was used.

AutomatioML Editor provides a graphical interface to the engineers in order to construct the model of the manufacturing environment. Furthermore, it has two additional interfaces. The first is the AutomationML2Tb which converts the AutomationML schema to the model that is understandable by the IoT Platform (ThingsBoard) and sets up the latter through its configuration API. The second, named AutomationML2OPCUA, translates the AutomationML to an XML for OPC UA description. This description is sent to the OPC UA server and to the virtual environment as a configuration file.

OPC UA Server acts as the communication gateway between all the industrial equipment, both real and virtual, and the IoT Platform. It is a classic OPC UA server which is configured to operate according to the model described by AutomationML, assuming that this model represents the manufacturing environment. OPC UA Server is also used as an abstraction for the connection of legacy systems (mainly PLCs) allowing their integration to the overall system.

Unity 3D Engine is the core system of the Virtual Manufacturing Environment. During the configuration phase it is responsible to construct the 3D Virtual Environment according to the AutomationML model. Additionally, it creates virtual interfaces to the MQTT interface of the IoT platform and also to a software PLC, emulating the communication of the IoT platform with the industrial and IoT equipment. During the execution phase the virtual environment is executing the 3D emulator based on its communication with the OPC UA server in two modes: a) active, which simulates the operation of the described production environment and b) passive, which duplicates the operation of the real production environment, by reading the OPC UA Server variables.

The proposed architecture is designed in order to provide flexibility and scalability. In case of a manufacturing shop floor with dozens of machines and controllers, it is feasible to have many UPC UA Servers. In the modelling procedure this can be achieved with different AutomationML files defining different namespaces, each one for a UPC UA Server. Furthermore, the ThingsBoard, which is the selected as IoT Platform, can be run in a cluster of virtual systems, allowing it to grow according to the connected devices/servers for supporting a large shop floor.

B. Off-line Configuration

As discussed previously, the flexibility of AutomationML for modelling purposes is not necessarily matched by other components selected for the needs of a system. In our proposed architecture, we use the Thingsboard platform to collect all data from industrial components and IoT devices. However, this platform offers only two types of entities: Assets and Devices. The former is an abstract concept that describes a logical grouping of devices or other assets, which can have some data attributes, but are not associated with telemetry data. The typical use case for these is to represent functional spaces (e.g. a "store room" asset, which can contain a "shelves" asset, and multiple devices such as "CO_2 sensor", "temperature sensor" etc). Without knowledge of this concept, the design task using AutomationML may result in models that are not compatible with this type of thinking. For example, in

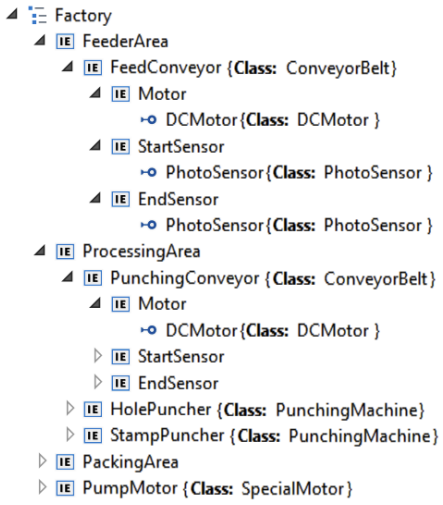


Fig. 2. Example AML Instance Hierarchy

AutomationML, a "store room" could well be a single System Unit Class object, with attributes for the CO_2 and temperature readings and without the need for these two sensors to be modelled as separate (individual) entities. Other concepts in AutomationML such as Roles and Interfaces cannot be modelled at all in platforms like ThingsBoard, which support relationships between assets, but not object-oriented concepts such as encapsulation and abstraction.

The conversion process is less problematic between AutomationML and OPC UA. The latter aims at exposing collections of "objects" to clients, through an information model defined in XML. These objects (referred to as *nodes*) contain a range of attributes and are interconnected through *references*. This enhanced flexibility allows for a more natural mapping of AutomationML concepts to OPc–UA, and in fact a guide for the conversion of AutomationML CAEX data to OPC-UA XML was made available in 2016 [15]. However, we also have to consider here how the conversion process works for the Thingsboard platform, in order to ensure a level of commonality in the design semantics. Next, we outline how this is achieved. In fig. 2 we demonstrate an excerpt from a manufacturing topology defined in AML. In this example, the factory contains manufacturing areas represented as Internal Elements without any accompanying information. Further, logical groupings of components are also represented in the same way (e.g. the "Processing Area" of the factory as a logical grouping of complex components located together). Manufacturing equipment components and devices are represented as instances of System Unit Classes (i.e. vendor-specific equipment), which implement appropriate Interfaces. Each Interface specifies the data attributes that we are interested to capture in the system for each System Unit Class, irrespective of its vendor or other capabilities. Therefore, we apply a design restriction in the process of generating the AML, where if an internal element does not implement an Interface, it means that it is used merely for logical grouping purposes (we term such

Design element	**AML equivalent**	**ThingsBoard equivalent**	**OPCXML Equivalent**
Physical equipment instance	SUC implementing Interface	Device	Object
Logical grouping of equipment	SUC without Interface	Asset	Folder
Equipment attributes	Interface attributes	Device additionalInfo property	Distinct variable or property node under Object

TABLE I
CONVERSION RULES BETWEEN AML, THINGSBOARD AND OPC

AML entities as *Groupings*). In cases where an Interface is implemented, then an operating component is being modelled (termed as *Devices*).

To proceed with the conversion of the instance hierarchy into the Thingsboard and OPC-UA platform, we developed converter written entirely in Javascript, which loads and parses the AML document, making use of recursive functions to drill into the hierarchy and identify individual components and their linkages to parent components (Fig. 3). The user simply selects an AML file from their computer, uploads it to the tool, and then clicks "Parse AML". Once the parsing is complete without errors, the user can click on the "Generate" buttons in the OPCUA-XML or Thingsboard sections to generate the relevant transformations.

At parsing time, information obtained from each AML node is stored into a memory-held linked list of objects. These in-memory objects are defined by a class that holds all the necessary information required to generate the related Thingsboard or OPCUA XML entities in the conversion, i.e. internal ID, parent object ID, AML-ID, OPC-ID, Thingsboard-ID, object name, object type (*Device* or *Grouping*) and attributes and their respective data types (held as JSON structures). Thingsboard-ID and OPC-ID are initially null at the time of parsing, and their values are updated as the user generates the relevant conversions from the UI. The basic conversion rules are shown in Table I. During parsing, *Groupings* are added to the Thingsboard platform as "Assets" and to OPCUA XML as "Folders". Similarly, *Devices* are added to both platforms' corresponding "Device" and "Object" types respectively. For Thingsboard, the "type" of each Asset is statically set to "Factory Component", while for devices, it is set to the name of the Interface it implements. We use the same approach to define the "description" element of Folder and Object entities in OPCUA-XML. Interfaces determine the data attributes which we wish to capture, however Thingsboard does not afford a method for specifically defining these during design (instead, attributes can be added as time-series in a completely ad-hoc manner, at any time). Our approach here is to store this information for each Device into the Thingsboard "additionalInfo" property. This property, provided by the ThingsBoard platform for each Device type, allows the storage of string data values. As such, we convert the attribute set of each AML Interface into an appropriate JSON-formatted string value, and

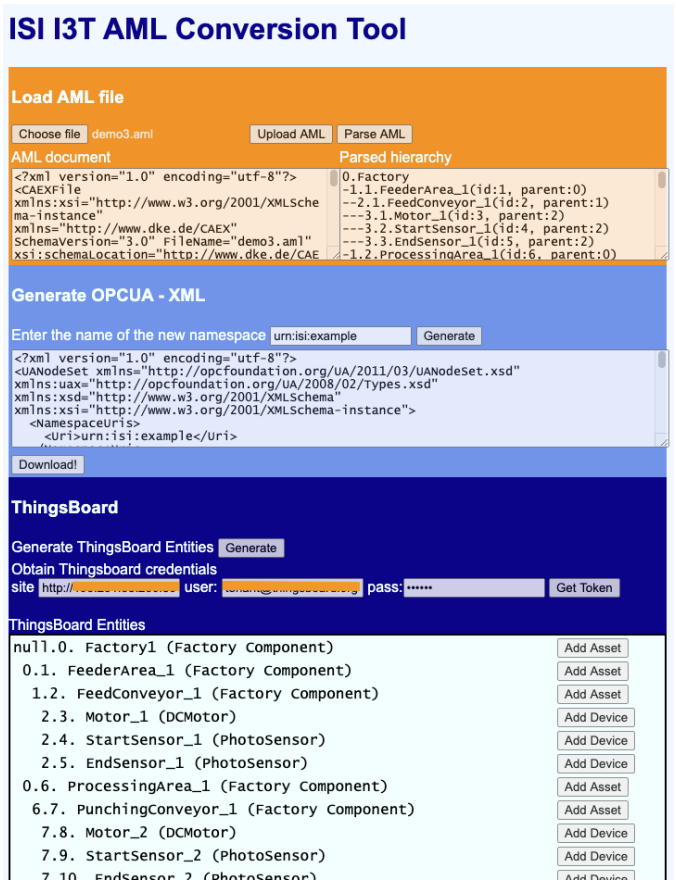


Fig. 3. Screenshot of the AML to TB/OPC converter UI. The user uploads an AML file and the converter parses the hierarchy, making the necessary suggestions about which elements to add and under what type (Asset or Device)

store them in the ”additionalInfo” property. This allows us to be able to programmatically acquire the attribute names defined in AML, and use them as timeseries data keys in Thingsboard, ensuring consistency between the AML model and the Thingsboard representation, and also enforcing data integrity checks during insertion of new data. On the other hand, OPCUA-XML requires the explicit definition of data attributes at design time. As such, we use the information from AML Interfaces, to directly specify equipment attributes as distinct Variable or Property nodes under each Object (including their data type).

Finally, with regard to maintaining the hierarchical relationships between AML objects, these are both supported by Thingsboard and OPCUA-XML. Since we store each entity’s Parent object ID at parsing, it is easy to replicate the relationships in both conversion processes, using the ”Relations” constructs in Thingsboard and ”HasComponent” node attributes in OPCUA-XML.

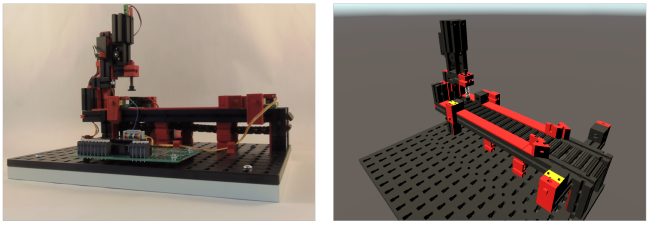
C. Execution Environment

The purpose of this architecture can be summarized as the information and communication integration of heterogeneous entities in a manufacturing environment. This is accomplished through the adaption of the AutomationML and OPC UA standards respectively. It can be stated that the first can be classified as a standard of the information layer of RAMI 4.0 and the second as a standard of the communication layer. Therefore, we have chosen to combine those two by integrating AutomationML in OPC UA, thus using the Automation ML for creating information models and communicating these models via the OPC UA protocol. As described in the previous paragraph, AutomationML is mostly used during the off-line configuration phase. However, OPC UA is used during the operating phase and constitutes the communication backbone of the architecture. All the information described by the AutomationML is exposed through OPC UA servers, which collect data from various sources, real or virtual.

V. Evaluating in a Hybrid Manufacturing Environment

A. Layout

For evaluating our approach we used a device from fischertechnik as a physical device (fig. 4(a)). The device consists of a conveyor for the transportation of the products and a punching machine for stamping them.



(a) (b)

Fig. 4. (a) The punching machine and (b) its virtual duplicate

The electrical signals from/to the machine (cables) are connected to an Arduino-based I/O slave, which converts them to Modubs TCP packets and sends them to OpenPLC, which is an IEC 61131-3 compliant open source industrial controller. The PLC is hosted on a raspberry pi and it is managed via a web interface. OpenPLC is accompanied by a Development environment for creating programs consisting of Program Organization Units (POU). Each POU can be programmed in each of the five available IEC 61131-3 languages. The controller supports Modbus protocol on top of TCP/IP stack.

The virtual manufacturing environment is implemented inside the Unity Game Engine. Everything inside the Unity’s world is called Gameobject and it can be created, accessed and edited via standard properties called Components. The Components, which are attached to Gameobjects, define their functional behaviour (fig. 4(b)). Prefabs are reusable assets which can be used for the instantiation of Gamobjects before or during runtime. We have already developed a prefab for the punching machine by constructing its 3D model and its animations, importing it into Unity and attaching the proper Components for simulating its physical behaviour. The behaviour of both the GameObjects and the Development

Environment can be further extended by writing C# scripts. In addition, we have implemented an AutomationML parser for importing and processing the AutomationML file and for instantiating the proper Gameobjects.

B. Scenario

The aim of the scenario executed for the validation of the aforementioned architecture is to demonstrate the success of the integration of various manufacturing entities, both real and virtual, in a way that ensures the integrity of the communication and information. Therefore, we used the AutomationML Editor to produce an XML-based description of a physical layout and afterwards this description was used by different entities for, on different levels, for generating useful products. Each entity interpreted the content of the description though its own view and level of understanding. The IoT platform created devices and assets, the AutomationML2OPCUA tool created the address space for the OPC UA server, and the AutomationML Parser inside Unity created GameObjects. The executed scenario can be described in more detail in the following steps.

- Creation of an AutomationML description of the manufacturing environment using standard and extended role class libs.
- Conversion of the AutomationML description to an XML based file compatible with OPC UA protocol.
- Launch of an OPC UA server that uses the XML based file of the previous step for generating its address space.
- Conversion of the AutomationML description to ThingsBoard's entities.
- The AutomationML description is imported to Unity.
- The Unity AutomationML parser searches for Instance Hierarchies (IEs) that implement a Role of an "OPC-UA Server" or a "ModbusPLC".
- The parser searches for IEs that have a specific Interface, which is called "Unity Interface" and then looks for children that implement the Role "OPCUAVar" or "ModbusVar". These roles have an attribute named refID which points to the corresponding server. Moreover, they hold details necessary for acquiring the data form the server, like the NodeID in the case of OPC UA.
- The "Unity Interface" has an attribute which holds the name of the Prefab that Unity has to instantiate.
- Finally, Unity instantiates the corresponding prefabs along with the necessary clients (Modbus or OPC UA).

VI. Conclusions

The paper deals with integration issues in the manufacturing environment. Integration challenges remain high as there is a large number of devices / equipment of high diversity that need to interoperate and participate in a common automation and control context. The advent of the Industrial Internet of Things has further enhanced this challenge.

The paper presents a proposed approach for the integration in the industrial environment utilizing AutomationML for shop floor modeling, OPC UA for service oriented information modeling, and an open source IoT platform as a core element. The approach also enrolls a virtual environment implemented via Unity 3D engine to enable factory floor emulation. The paper presents the overall system architecture followed and details both its configuration and run time execution. Some first evaluation of the system has been performed and a relevant scenario is presented.

Future work will be associated to further evaluation of the proposed approach, targeted to its evolution towards the Digital Twin concepts.

References

[1] F. Xia, L. T. Yang, L. Wang, and A. Vinel, "Internet of things," *International journal of communication systems*, vol. 25, no. 9, p. 1101, 2012.

[2] A. S. Lalos, A. P. Kalogeras, C. Koulamas, C. Tselios, C. Alexakos, and D. Serpanos, "Secure and safe iiot systems via machine and deep learning approaches," in *Security and Quality in Cyber-Physical Systems Engineering*. Springer, 2019, pp. 443–470.

[3] Y. Lu, C. Liu, I. Kevin, K. Wang, H. Huang, and X. Xu, "Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101837, 2020.

[4] P. Adolphs, S. Auer, H. Bedenbender, M. Billmann, M. Hankel, R. Heidel, M. Hoffmeister, H. Huhle, M. Jochem, M. Kiele *et al.*, "Structure of the administration shell. continuation of the development of the reference model for the industrie 4.0 component," *ZVEI and VDI, Status Report*, 2016.

[5] S.-W. Lin, B. Miller, J. Durand, G. Bleakley, A. Chigani, R. Martin, B. Murphy, and M. Crawford, "The industrial internet of things volume g1: reference architecture," *Industrial Internet Consortium*, pp. 10–46, 2017.

[6] Y. Zhang, G. Zhang, J. Wang, S. Sun, S. Si, and T. Yang, "Real-time information capturing and integration framework of the internet of manufacturing things," *International Journal of Computer Integrated Manufacturing*, vol. 28, no. 8, pp. 811–822, 2015.

[7] F. Tao, J. Cheng, and Q. Qi, "Iihub: An industrial internet-of-things hub toward smart manufacturing based on cyber-physical system," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 5, pp. 2271–2280, 2017.

[8] C. Alexakos, A. Komninos, C. Anagnostopoulos, G. Kalogeras, A. Savvopoulos, and A. Kalogeras, "Building an industrial iot infrastructure with open source software for smart energy," in *2019 First International Conference on Societal Automation (SA)*. IEEE, 2019, pp. 1–8.

[9] S. Choi, K. Jung, B. Kulvatunyou, and K. Morris, "An analysis of technologies and standards for designing smart manufacturing systems," *J Res Natl Inst Stan*, vol. 121, pp. 422–433, 2016.

[10] S. Faltinski, O. Niggemann, N. Moriz, and A. Mankowski, "Automationml: From data exchange to system planning and simulation," in *2012 IEEE International Conference on Industrial Technology*. IEEE, 2012, pp. 378–383.

[11] R. Drath, A. Luder, J. Peschke, and L. Hundt, "Automationml-the glue for seamless automation engineering," in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE, 2008, pp. 616–623.

[12] A. Lüder, L. Hundt, and A. Keibel, "Description of manufacturing processes using automationml," in *2010 IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*. IEEE, 2010, pp. 1–8.

[13] G. N. Schroeder, C. Steinmetz, C. E. Pereira, and D. B. Espindola, "Digital twin data modeling with automationml and a communication methodology for data exchange," *IFAC-PapersOnLine*, vol. 49, no. 30, pp. 12–17, 2016.

[14] C. Koulamas and A. Kalogeras, "Cyber-physical systems and digital twins in the industrial internet of things [cyber-physical systems]," *Computer*, vol. 51, no. 11, pp. 95–98, 2018.

[15] "Din spec 16592: Combining opc unified architecture and automation markup language," *DIN Spec*, 2016.