Enhancing Interoperability of Digital Twins Based on Digital Twins Definition Language

Salvatore Cavalieri[a] and Salvatore Gambadoro[b]
Department of Electrical Electronic and Computer Engineering, University of Catania, Catania-95125, Italy

Keywords: Interoperability, Digital Twin, DTDL, OPC UA, Industry 4.0.

Abstract: The Industry 4.0 is featured by a continuously-evolving digital transformation, aiming to automate all the traditional industrial practices. Digital Twin is one of the most important solutions to reach this aim. Among the standards currently available to realize Digital Twins there is the Digital Twins Definition Language. Digital Twin requires exchange of data with the real system it models and with other applications which use the digital replica of the system. In the context of Industry 4.0, a reference standard for an interoperable exchange of information between applications, is Open Platform Communications Unified Architecture. The idea behind the paper is to exploit this standard to allow a Digital Twin based on Digital Twins Definition Language to exchange data with any applications compliant to the Open Platform Communications Unified Architecture. A proposal about the mapping from Digital Twins Definition Language to Open Platform Communications Unified Architecture will be presented and discussed in this paper.

1 INTRODUCTION

Industry 4.0 is featured by a continuously-evolving digital transformation, aiming to automate all the traditional industrial practices. Among the several solutions to reach this aim, there is the Digital Twin (DT), bringing as much of the equipment from the physical space into the virtual domain. Digital Twins emerged as an experimental technology set to enable replication of elements, functions, operations and dynamics of physical systems into digital world, with better control at testing, analysis, prediction and hazard prevention for sensitive processes (Mihai, 2022; Rocha, 2022).

Digital Twins are used in different industrial settings, including health surveillance, agriculture, smart cities, smart grids, manufacturing, meteorology, education and automobiles. Digital Twins support the development of production processes making them reliable and flexible, enabling to visualise, monitor and optimize processes (Mihai, 2022; Rocha, 2022; Rasheed, 2020).

Different organisations are currently working aiming to standardize Digital Twin definition, interoperability and how to interact with these Digital Twins. Two notable projects in this area are the Asset Administration Shell (AAS) (Plattform Industrie 4.0, 2020; Pribiš, 2021; Jacoby, 2020) and the Digital Twin Definition Language (DTDL) (DTDL, 2023). The DTDL was born as an open source initiative by Microsoft and it is already used in many commercial services offered by Microsoft like IoT Hub, IoT Central, and Azure Digital Twins (Jacoby, 2020; DTDL, 2023).

One of the main features of a Digital Twin is the communication with the physical world, from which a Digital Twin must receive the current state (e.g., collection of data measured by sensors). At the same time, a Digital Twin may have the need to exchange data with different applications in order to realize particular processes according to the aims to be reached (e.g., monitoring, testing, analysis, prediction, maintenance). Finally, the output data produced as a result of the processes done by the Digital Twin must be sent to the physical system for which it realizes a replica. For this reason, data exchange must be considered an important part of a Digital Twin; without data exchange, most of functions of a Digital Twin could not be realized (Qi, 2018). Interoperability of the data exchange seems a

[a] https://orcid.org/0000-0001-9077-3688
[b] https://orcid.org/0000-0001-9840-1379

Cavalieri, S. and Gambadoro, S.
Enhancing Interoperability of Digital Twins Based on Digital Twins Definition Language.
DOI: 10.5220/0011981000003467
In *Proceedings of the 25th International Conference on Enterprise Information Systems (ICEIS 2023) - Volume 1*, pages 741-748
ISBN: 978-989-758-648-4; ISSN: 2184-4992

very important requirement, allowing a Digital Twin to communicate with a multitude of physical systems and applications.

Interoperability between applications is considered one of the main goals of Industry 4.0 (Liao, Ramos, et ali. 2017; Liao, Deschamps, et ali. 2017; Lelli, 2019). Open Platform Communications Unified Architecture (OPC UA), is considered one of the main reference standards for an interoperable exchange of information between applications inside Industry 4.0 (Ladegourdie, 2022; Ferrari, 2018; González, 2019). The OPC UA is based on two communication models: client/server and publish/subscribe; a comprehensive information model allows to represent data and the relevant semantics.

The idea behind the paper is to enhance the interoperability of a Digital Twin through integration into the OPC UA domain. Figure 1 shows a Digital Twin exchanging data with the real system it models, and with applications using the DT. Data exchange with applications based on OPC UA communication system may be enabled through a solution able to map the entire set of information maintained by a Digital Twin into the OPC UA domain. Mapping should include every semantic aspect of the Digital Twin, in order to really enable an interoperable data exchange between the two domains. The proposal presented in this paper is based on the definition of a custom OPC UA information model able to realize this mapping. Figure 1 shows only an example of interoperability; in this example, the information model of the OPC UA Server shown by Figure 1, is able to represent each element of the Digital Twin in the OPC UA domain, making available the Digital Twin and its relevant content to whatever application based on OPC UA client role. The mapping solution here presented enables a Digital Twin to have a counterpart in the OPC UA domain; each information (including semantic) maintained in a Digital Twin can be accessed by a plethora of OPC UA-compliant applications.

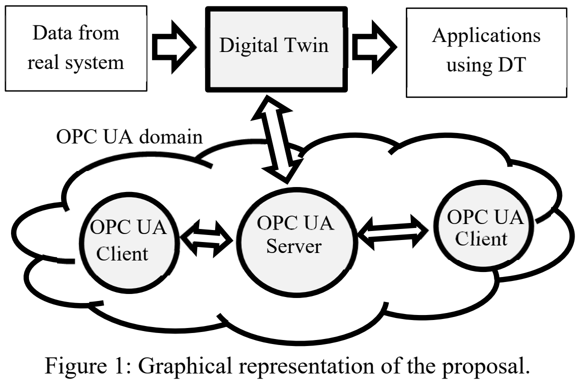

Figure 1: Graphical representation of the proposal.

Among the available Digital Twins, the Digital Twin Definition Language model will be considered in this proposal. The definition of a custom data structures in the OPC UA information model, able to represent each element of the original DTDL-based Digital Twin, will be introduced in this work. The proposed mapping has been implemented in order to be validated; the relevant implementation and validation will be introduced in the paper.

2 RELATED WORK

Current literature provides a lot of publications about the use of OPC UA information model to structure and expose data coming from different domains of interest in order to achieve fully interoperability.

Considering mapping between Digital Twin and OPC UA, several proposals are also present in the current literature. Mapping of AAS to OPC UA is proposed by (Cavalieri, 2020). Another example of integration of AAS with OPC UA is given by (Arm, 2021). Integration of AAS and OPC UA is also subject of the official specifications (Plattform Industrie 4.0, 2020; OPC 30270, 2023), which define an OPC UA model to expose AAS information to OPC UA applications.

To the best of authors' knowledge, considering the integration of DTDL with OPC UA, only the software solution, called OPCUA2DTDL (OPCUA2DTDL, 2023), is currently available. This solution aims to convert an OPC UA information model into DTDL constructs; a DTDL digital model can be built starting from its definition based on OPC UA specification. The main limit of this solution is that mapping in the opposite direction is not allowed. In other terms, given an already defined DTDL-based Digital Twin is not possible to achieve its counterpart in the OPC UA domain. Introduction pointed out that the paper has this aim, proposing an integration able to map a Digital Twin realised by DTDL into OPC UA information model. This is a very important consideration which allows to point out the originality of the proposed solution made in this paper.

3 OPC UA INFORMATION MODEL

The OPC UA provides a semantically enriched information model in order to represent data. The OPC UA Information Model allows a server to expose information to clients; publish/subscribe communication model is based on the same information model used for the client/server communication model.

The OPC UA Information Model is organized by OPC UA Nodes grouped together to compose the so-called OPC UA AddressSpace (Mahnke, 2009; OPC Unified Architecture Part 3, 2023; OPC Unified Architecture Part 5, 2023).

Each OPC UA Node belongs to a class named NodeClass. In (OPC Unified Architecture Part 3, 2023; OPC Unified Architecture Part 5, 2023) it is possible to have detailed information about the OPC UA NodeClasses.

OPC UA defines standard graphical representation for OPC UA NodeClasses. The reader should refer to Annex C of (OPC Unified Architecture Part 3) to have a complete description of this representation, which will be used in this paper.

4 DTDL

The Digital Twins Definition Language (DTDL) (DTDL, 2023) is a formalism capable of describing Digital Twin devices and assets. DTDL uses JSON-LD (JSON-LD, 2023).

According to DTDL, the structure and behaviour of a Digital Twin is fully described by six classes of metamodels: Interface, Telemetry, Property, Control, Relationship and Component.

Every resource is modelled by an interface which can contain a set of telemetry, properties, commands, relationship and components. Telemetry describes data emitted by a resource, whether it is a regular stream of sensor readings or a calculated data stream; Telemetry does not store any data.

Properties define values within a Digital Twin; these values can be read-only or have read and write states. Properties have a backing storage.

Commands correspond to functions that can be invoked with optional input and output parameters. Relationship describes a link to another digital twin and makes it possible to create graphs of digital twins. Component models the entities that exist in the DT, including sensors, gateways, and digital systems.

5 MAPPING DTDL TO OPC UA

As it was pointed out in Section 2, literature provides a lot of publications featuring the use of OPC UA Information Model to structure and expose information coming from different domains of interest. The reader may refer to (Cavalieri, Salafia, 2020) to have an overview of the common practices adopted when OPC UA Information Model is used to model a generic system. In this proposal, these common practices have been taken into account to map the DTDL metamodel classes into OPC UA Information Model; in particular, custom OPC UA types have been defined through an extension of the current OPC UA Information Model, able to represents the DTDL metamodel classes and the relevant properties.

The following subsections will present the mapping of the six DTDL metamodels classes, introducing the custom OPC UA types defined to realize the proposal.

5.1 Interface

It has been assumed to map the Interface class with a custom OPC UA ObjectType, called DTDLInterfaceType. Some of the attributes of this type were used to represent a subset of the properties of the Interface class. Other properties of Interface class were mapped using OPC UA Nodes connected to the OPC UA DTDLInterfaceType ObjectType, as shown by Figure 2.

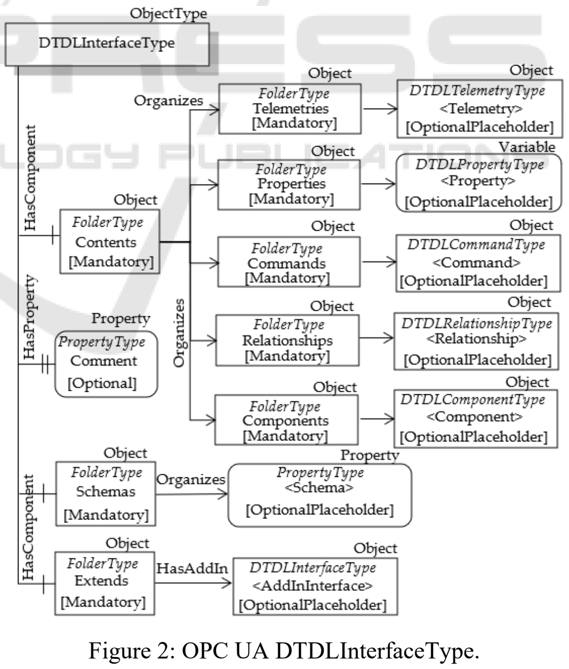

Figure 2: OPC UA DTDLInterfaceType.

Among the DTDL properties of the Interface class there is the property “contents”, which allows to define a set of objects representing the contents of an Interface; these objects belong to the other DTDL classes (i.e., telemetry, properties, commands, relationship and components). In order to map this set of objects, a particular folder (named Contents in Figure 2) is used. It contains several other folders (i.e. Telemetries, Properties, Commands, Relationships and Components), each of which organizes the OPC UA Nodes used to represent the other DTDL classes; the relevant mappings will be described in the following subsections.

Another DTDL property is called "schemas"; it represents the set of reusable data types which are used in a digital twin interface. In OPC UA the DataType NodeClass may be used to define the representation of the DTDL data types. The OPC UA FolderType Object called Schemas, shown in Figure 2, has been used to organize the OPC UA Property Nodes, each containing the description of the OPC UA DataType mapping the DTDL schemas featured by the current interface.

Another folder present in Figure 2 is the Extends FolderType Object. It maps the "extends" property of the Interface Class. In DTDL, Interfaces can inherit from multiple interfaces; for this reason, the DTDL "extends" property is used to maintain details of the set of interfaces each interface inherits from. As shown by Figure 2, the use of a particular reference, called HasAddIn, allows the Extends folder to point to the DTDLInterfaceType Objects modelling the DTDL Interfaces the current interface inherits from.

5.2 Telemetry

The metamodel class Telemetry describes the data emitted by any digital twin. Telemetry does not store any data.

Due to the lack of data stored by a DTDL Telemetry element, it has been assumed to map it with a custom OPC UA ObjectType, called DTDLTelemetryType. Some of its basic attributes were used to represent a subset of the properties of the DTDL Telemetry class. Other properties were mapped by OPC UA Nodes as shown by Figure 3.

As it can be seen, the DTDLTelemetryType ObjectType has two optional properties, Comment and Unit, mapping the Telemetry Class properties "comment" (which allows to define a comment for model authors) and "unit" (which defines a semantic type of the Telemetry), respectively.

The DTDL "schema" property represents the data type of the Telemetry; OPC UA DataType seems suitable to represent this property, as said before. In other to have a mapping from DTDL to OPC UA domain, the OPC UA Property Node, called Schema in Figure 3, is considered; it contains the description of the OPC UA DataType mapping the DTDL schemas featured by the current property.

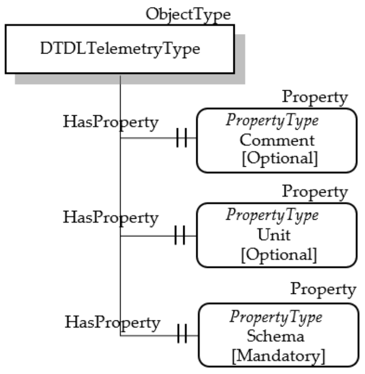

Figure 3: OPC UA DTDLTelemetryType.

5.3 Property

The metamodel class Property allows to store values within a digital twin. These values can be read-only or have read and write states. For example, a device serial number may be a read-only value that can be read at any time; the desired temperature on a thermostat may be a read-write value that can be updated.

As DTDL Property class is used to store values, it seems that it may be represented very well by the OPC UA DataVariable Node. For this reason, it has been assumed to map the DTDL Property class with a custom OPC UA DataVariableType, called DTDLPropertyType. Some of the basic attribute of the DTDLPropertyType were used to represent properties of the Property class. Other properties have been mapped as shown by Figure 4. The DTDLPropertyType VariableType features two optional properties, Comment and Unit, mapping the DTDL Property class properties "comment" and "unit", respectively.

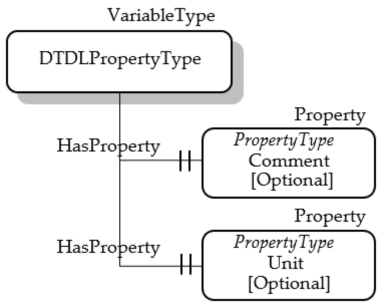

Figure 4: OPC UA DTDLPropertyType.

It is important to point out that the proposed modelling of DTDL Property class with the custom DTDLPropertyType DataVariableType, allows to make available the OPC UA attribute Value, which may be used to contain the real value of the digital twin.

5.4 Command

The metamodel class DTDL Command allows to describe a function or operation that can be performed on digital twins.

It was assumed to represent the Command class through a custom OPC UA ObjectType, called DTDLCommandType, featuring the structure shown by Figure 5.

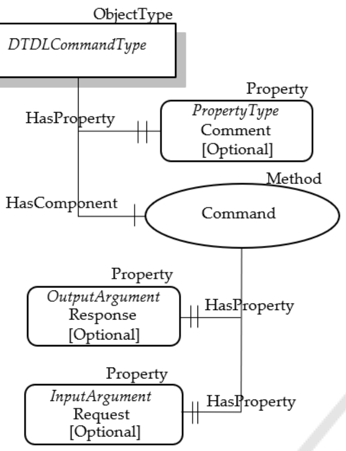

Figure 5: OPC UA DTDLCommandType.

As shown, the DTDLCommandType ObjectType features a property and a method. The property called Comment models the DTDL "comment" property of the DTDL Command class. The OPC UA method called Command represents the function/operation modelled by the DTDL Command class; it features optional input and output parameters (called Request and Response, respectively).

5.5 Relationship

The metamodel class DTDL Relationship describes a link to another (separate) digital twin and enables graphs of digital twins to be created.

As done for other DTDL metamodel classes, this class has been mapped using an OPC UA ObjectType. A custom ObjectType has been defined and called DTDLRelationshipType, shown by Figure 6. In particular, it features the three properties Comment, MinMultiplicity and MaxMultiplicity. They model the DTDL properties "comment", "minMultiplicity" and "maxMultiplicity", respectively.

The DTDL property named "properties" represents the set of elements of Property class that define relationship-specific state. In order to represent this set, a folder has been considered in OPC UA; in the Figure 6 it is called Properties. This folder will contain OPC UA Nodes each modelling an element of Property class, using OPC UA Property Nodes belonging to DTDLPropertyType.

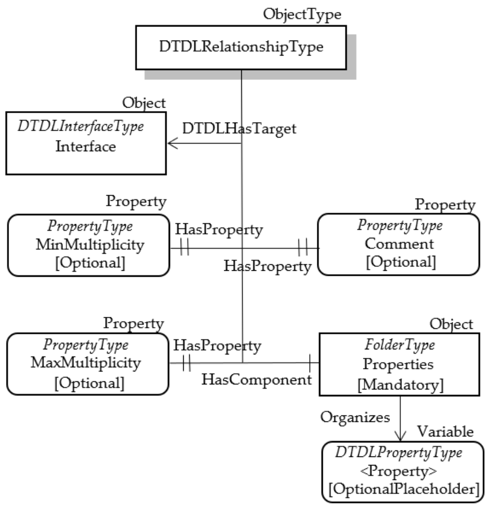

Figure 6: OPC UA DTDLRelationshipType.

The last DTDL property to be mapped is the "target", representing the link to a metamodel class Interface. In OPC UA links between Nodes are realized using References; for this reason, the DTDL "target" was mapped into OPC UA through a Reference pointing to the OPC UA Node modelling the Interface to which the Relationship refers. A custom Reference, called DTDLHasTarget has been defined.

5.6 Component

The metamodel class Component enables interfaces to be composed of other interfaces.

A custom OPC UA ObjectType called DTDLComponentType has been defined, as shown by Figure 7.

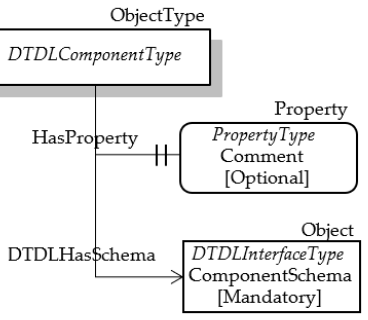

Figure 7: OPC UA DTDLComponentType.

It has an optional property called Comment, modelling the DTDL property "comment". The other

DTDL property to be mapped is the "schema", which defines the DTDL Interface of the component. As the DTDL Interface has been modelled in OPC UA by an Object of DTDLInterfaceType type, Figure 7 shows the presence of a mandatory Object of this type. In order to allow the DTDLComponentType Object to point to the DTDLInterfaceType Object, a custom Reference has been defined called DTDLHasSchema, as shown by the Figure 7.

6 IMPLEMENTATION

The custom OPC UA Information Model presented in the previous section has been implemented by the authors and the software implementation is freely available at the GitHub repository (DTDL-OPCUA-Information-Models-Mapping, 2023).

7 CASE STUDY

Using the software implementation, the proposal has been validated considering several case studies. In the following subsections details about the different steps followed in the validation procedure will be given. Details will refer to a very simple scenario, considered in order to be easily read and understood.

It has been assumed to take into consideration a Digital Twin of a building. In particular, a basic model of a room has been defined and its extension to a meeting room has been considered. Figure 8 shows the description of the DT model called "Room", modelling a room; the model is made up by the DTDL Interface "Room" featuring only one Property (i.e. "setlight").

```
{
  "@id": "dtmi:example:Room;1",
  "@type": "Interface",
  "@context": "dtmi:dtdl:context;2",
  "displayName": "Room",
  "contents": [
    {
      "@type": "Property",
      "name": "setlight",
      "schema": "boolean",
      "writable": true
    }
  ]
}
```

Figure 8: DTDL model of a room.

Another DTDL model has been considered, called "MeetingRoom"; it models a meeting room and includes all the properties of the "Room" model, adding other ones. As shown by Figure 9, the DTDL Interface called "MeetingRoom" features a contents property made up by two Property elements (i.e., "occupied" and "tempValue"). This interface extends the simpler interface called "Room".

```
{
 "@id": "dtmi:example:MeetingRoom;1",
 "@type": "Interface",
 "@context": "dtmi:dtdl:context;2",
 "extends": [ "dtmi:example:Room;1" ],
 "displayName": "MeetingRoom",
 "contents": [
     {
         "@type": "Property",
         "name": "occupied",
         "schema": "boolean",
         "writable": true
     },
     {
         "@type": "Property",
         "name": "tempValue",
         "schema": "double",
         "writable": true
     }
 ]
}
```

Figure 9: DTDL model of a meeting room.

7.1 Digital Twin in Azure Platform

The DTDL model "MeetingRoom" shown by Figure 9 was implemented inside Microsoft Azure Digital Twin platform available at (Microsoft Azure, 2023). A particular tool named "Azure Digital Twin Explorer" is available in this platform to import DTDL models and to create instance of Digital Twins based on DTDL. The "Room" and "MeetingRoom" DTDL models were uploaded in this tool, and a single instance of the "MeetingRoom" Interface has been created; the instance was named "MeetingRoomA".

7.2 OPC UA Server

The next step of the proposed mapping is the definition of an OPC UA Server able to map each Digital Twin instance into the OPC UA Information Model. In our simple case study, the Digital Twin instance is the "MeetingRoomA".

OPC UA Server was implemented in NodeJS using the NodeOPCUA (NodeOPCUA, 2023), which is an OPC UA SDK making available the OPC UA communication stack written in TypeScript for NodeJS.

According to the mapping solution presented in Section 5, the OPC UA AdressSpace mapping the "MeetingRoomA" instance may be achieved, as shown by Figure 10. The OPC UA Object

"MeetingRoomA" is an instance of the DTDLInterfaceType ObjectType. It features the folder Contents, containing the folder Properties, which organizes two OPC UA DataVariable Nodes instances of the DTDLPropertyType type. They model the two DTDL properties "occupied" and "tempValue", shown by Figure 9. Figure 10 points out the main attributes of these Objects. The OPC UA Object "MeetingRoomA" features another component made up by the mandatory folder Extends; the HasAddIn reference allows this folder to point to the DTDLInterfaceType Object modelling the DTDL Interfaces the current interface inherits from, i.e. the Interface named "Room". This interface is modelled by the OPC UA DTDLInterfaceType Object "Room", featuring one OPC UA DataVariable Node organized by the Properties folder. This Node represents the DTDL property named "setlight" shown in Figure 8.

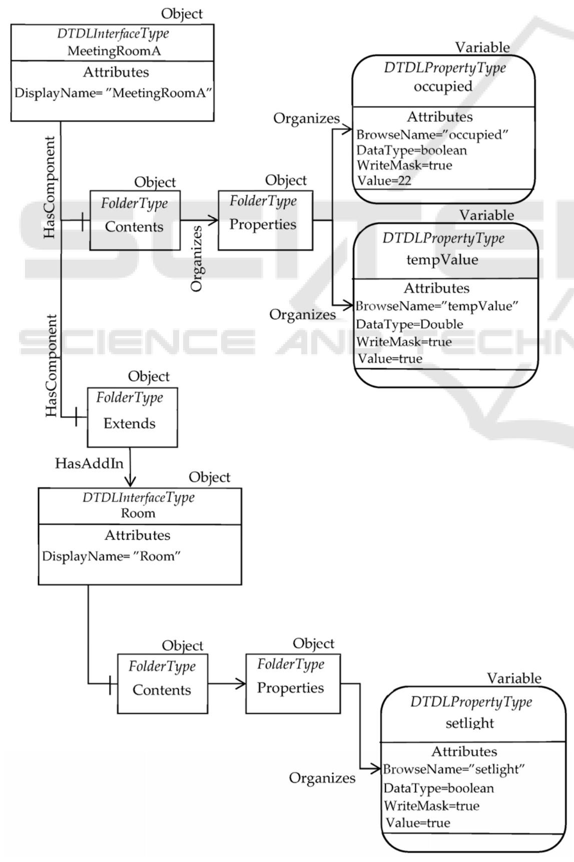

Figure 10: Mapping the instance of the meeting room model using the OPC UA DTDL InterfaceType.

The OPC UA AddressSpace shown by Figure 10 was implemented using the UaModeler (UaModeler, 2023) tool. Figure 11 shows the graphical representations of the OPC UA Nodes that have been created using the UaModeler tool.

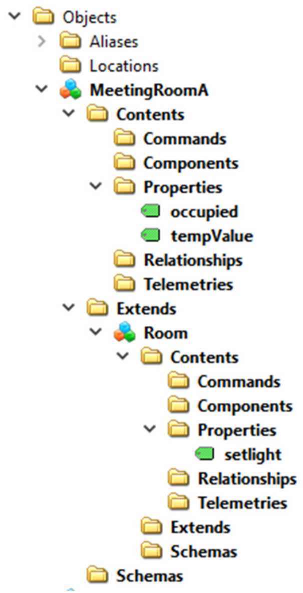

Figure 11: OPC UA Nodes created for this case study.

Data values maintained by a Digital Twin instance changes over the time due to updates of the real system for which the Digital Twin realizes a digital replica. It is clear that information maintained by the Digital Twin instance and by the OPC UA Server must be consistent. This means that each change in a property of the Digital Twin instance must be updated into the OPC UA Server, and vice versa. For this reason, a custom program was implemented inside the server in order that each time an information changes inside the Digital Twin instance, the same change must be reflected in the OPC UA Server, and vice versa. This code has been developed in NodeJS, as said; the Azure SDK for NodeJS (azure-sdk-for-js, 2023) has been also used; this software realizes the communication stack needed by a NodeJS program to the access to the Digital Twin instances.

7.3 Validation

The last step followed in the validation procedure adopted by the author was the execution of several tests aimed to verify the consistency between information maintained by Digital Twin instance and OPC UA AddressSpace. Considering the simple case study here presented, several changes in the properties of the DT instance "MeetingRoomA" were performed, verifying that the consistent changes occurred in the OPC UA Server, accessing the information by OPC UA client applications.

8 CONCLUSIONS

The paper has introduced a solution of mapping from DTDL to OPC UA Information Model. The proposal allows to enable interoperability from DTDL-based Digital Twins and OPC UA. The article is original as the issue has not been dealt with so far. The implementation of the mapping here proposed, has been done by the authors in order to demonstrate the feasibility of the proposed solution.

REFERENCES

azure-sdk-for-js (2023). Available online: https://github.com/Azure/azure-sdk-for-js/tree/%40azure/digital-twins-core_1.1.0/sdk (accessed on 24 February 2023).

Arm, J., Benesl, T., Marcon, P., Bradac, Z., Schröder, T., Belyaev, A., Werner, T., Braun, V., Kamensky, P., Zezulka, F., Diedrich, C., Dohnal, P. (2021). Automated Design and Integration of Asset Administration Shells in Components of Industry 4.0. Sensors, 21(6), 2004.

Cavalieri, S., Salafia, M.G. (2020) Insights into Mapping Solutions Based on OPC UA Information Model Applied to the Industry 4.0 Asset Administration Shell. Computers, 9, 28.

DTDL (2023). Digital Twins Definition Language. Available online: https://github.com/Azure/opendigitaltwins-dtdl/blob/master/DTDL/v2/dtdlv2.md (accessed on 24 February 2023).

DTDL-OPCUA-Information-Models-Mapping (2023). Available online: https://github.com/OPCUAUniCT/DTDL-OPCUA-Information-Models-Mapping (accessed on 24 February 2023).

Ferrari, P., Flammini, A., Rinaldi, S., Sisinni, E., Maffei, D., Malara, M. (2018). Impact of Quality of Service on Cloud Based Industrial IoT Applications with OPC UA. Electronics, 7(7), 109.

González, I., Calderón, A.J., Figueiredo, J., Sousa, J.M.C. (2019). A Literature Survey on Open Platform Communications (OPC) Applied to Advanced Industrial Environments. Electronics, 8(5), 510.

Jacoby, M., Usländer, T. (2020). Digital Twin and Internet of Things—Current Standards Landscape. Applied Sciences, 10(18), 6519.

JSON-LD 1.1-A JSON-based Serialization for Linked Data. Available online: https://www.w3.org/TR/json-ld11/ (accessed on 24 February 2023).

Ladegourdie, M., Kua, J. (2022). Performance Analysis of OPC UA for Industrial Interoperability towards Industry 4.0. IoT, 3(4), 507–525.

Lelli, F. (2019). Interoperability of the Time of Industry 4.0 and the Internet of Things. Future Internet, 11(2), 36.

Liao, Y., Deschamps, F., Loures, E.F.R., Ramos, L.F.P. (2017). Past, present and future of Industry 4.0 - a systematic literature review and research agenda proposal, International Journal of Production Research, 55:12, 3609-3629

Liao, Y., Ramos, L.F.P., Saturno, M., Deschamps, F., Loures, E.F.R., Szejka, A.L. (2017) The Role of Interoperability in The Fourth Industrial Revolution Era,IFAC-PapersOnLine,50(1)12434-12439.

Mahnke, W., Leitner, S.H., Damm, M. (2009). OPC Unified Architecture; Springer.

Microsoft Azure (2023). Available online: https://portal.azure.com/#home (accessed on 24 February 2023).

Mihai, S., et al. (2022). Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects. IEEE Communications Surveys and Tutorials, 24(4), 2255-2291.

NodeOPCUA (2023). Available online: https://node-opcua.github.io/ (accessed on 24 February 2023).

OPC 30270 (2023). OPC UA for I4 Asset Administration Shell. Available online: https://opcfoundation.org/developer-tools/documents/view/273 (accessed on 24 February 2023).

OPCUA2DTDL (2023). OPC UA to DTDL Conversion Tool. Available online: https://github.com/khilscher/OPCUA2DTDL (accessed on 24 February 2023).

OPC Foundation. OPC Unified Architecture Part 3: Address Space Model. Available online: https://opcfoundation.org/developer-tools/documents/view/160 (accessed on 24 February 2023).

OPC Foundation. OPC Unified Architecture Part 5: Information Model. Available online: https://opcfoundation.org/developer-tools/documents/view/162 (accessed on 24 February 2023).

Plattform Industrie 4.0 (2020), ZVEI. Details of the Asset Administration Shell—Part 1. Available online: https://www.zvei.org/en/press-media/publications/details-of-the-asset-administration-shell/ (accessed on 24 February 2023).

Pribiš, R., Beňo, L., Drahoš, P. (2021). Asset Administration Shell Design Methodology Using Embedded OPC Unified Architecture Server. Electronics, 10(20), 2520.

Qi, Q., Tao, F. (2018). Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. IEEE Access, 6, 3585–3593.

Rasheed, A., San, O., Kvamsdal, T. (2020). Digital Twin: Values, Challenges and Enablers. IEEE Access 2020, 8, 21980–22012.

Rocha, H.d., Pereira, J., Abrishambaf, R., Espirito Santo, A. (2022). An Interoperable Digital Twin with the IEEE 1451 Standards. Sensors, 22(19), 7590.

UaModeler (2023). Available online: https://documentation.unified-automation.com/uamodeler/1.3.0/html/index.html (accessed on 24 February 2023).