ISBN: 978-981-18-5852-9
WCSE 2022 Spring Event: 2022 9th International Conference on Industrial Engineering and Applications
doi: 10.18178/wcse.2022.04.092

Design and Validation of an Internet of Things (IoT) based Architecture for the Construction Industry

Zhen Cai [1+], Deniz Etkar [2], Stephan Kessler [1] and Johannes Fottner [1]

[1] School of Engineering and Design, Technical University of Munich, Garching/Munich, Germany

[2] Department of Informatics, Technical University of Munich, Garching/Munich, Germany

Abstract. Logistics plays a significant role in construction projects. However, same as other processes in this industry, it is poorly digitalized and automated. Internet of Things (IoT) has been proved to be a good solution for digitization and automation in many industrial cases, but its potential in the construction industry is not fully exploited. This paper aims to conceptualize a cloud-based IoT architecture to meet the requirements of involved parties in construction logistics. The requirements include using diverse IoT systems and implementing real-time communication between devices and machines with OPC Unified Architecture (OPC-UA). As validation, the architecture is implemented in the Microsoft Azure infrastructure with use cases of IoT devices and OPC-UA devices. A concept of Role-Based Access Control (RBAC) is also implemented to ensure the data security. Based on this architecture, construction logistics can be digitalized through various IoT technologies with data consistency, security, and flexibility, thus increasing the workflows efficiency in the construction industry.

Keywords: Architecture; IoT; OPC-UA; Role-Based Access Control; Digitization; Construction Logistics; Microsoft Azure

1. Introduction

The construction industry has been one of the least digitized and automated industries for decades [1], which has led to low productivity across the board [2]. However, as technology in construction industry evolves, the complexity of construction projects overwhelms traditional ways of working, leading to an increasing need for digitization and automation in the industry [3]. This trend has led to new development topics in various aspects of the construction industry, such as Building Information Modelling (BIM) in the planning phase, on-site robotics in the construction phase, and Internet of Things applications in the construction logistics [4]. International Telecommunication Union (ITU) defines the Internet of Things (IoT) as “a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies” [5], which fits right in with the demand of digitization and automation in the construction industry. Its potential of improving efficiency of the construction processes has been highly recognized by the academic community [6].

However, in practice, the effectiveness of IoT applications in construction logistics is still limited for various reasons. First, a construction project involves many constructions machinery and materials [7], resulting in the diversity of required IoT systems. For example, the construction machinery needs high-level edge devices, whereas the materials could be tracked by scanning QR-Codes. Moreover, the use of different systems also leads to difficulties in information exchange, as data is prosecuted and stored separately in the IoT providers’ systems. Finally, construction logistics has involved a lot of related parties, e.g., construction companies, construction machinery manufacturers, IoT vendors, materials suppliers, which places high requirements on information systems hierarchies [8].

The purpose of this paper is to conceptualize a cloud-based IoT architecture for construction logistics to meet their needs for digitization and automation. The next chapter presents the research of existing IoT architectures and projects. At the same time, the related parties in the construction logistics are interviewed

[+] Corresponding author. Tel.: + 49 89 289 15423; fax: + 49 89 289 15922.
E-mail address: zhen.cai@tum.de.

to obtain requirement analysis from a different perspective. Afterward, an architecture design is proposed based on literature research and requirements analysis results. This architecture is then implemented in the Microsoft Azure cloud and validated with use cases. Finally, the architecture is evaluated, and recommendations for future work are presented.

2. Related Work and Requirements Analysis

Since the origin of the IoT concept, the development of related technologies has been flourishing for decades [9]. As the technology has evolved, different IoT architectures have also been developed. Firstly, presented in 2013, the IoT Architectural Reference Model (IoT ARM) is a guideline to design and develop IoT systems [10]. The IoT ARM first introduces the IoT Domain Model, which defines the technical details of the four core aspects of an IoT system: information, functionality, communication, and security. The technology model is then abstracted to the architectural level (viewpoints) by adding information such as functional requirements from stakeholders. In addition, non-functional requirements are also covered in the perspectives in the IoT ARM [10]. In 2015, the Industrial Internet of Things Consortium (IIC) had proposed the Industrial Internet Reference Architecture (IIRA). Like the IoT ARM, the IIRA is defined in viewpoints: business, usage, function, implementation. The difference between these two reference architectures is that IoT ARM focuses on the information model while IIRA focuses more on operations and

applications [11]. Another essential concept is the Reference Architectural Model Industry 4.0 (RAMI4.0), which was standardized in 2016 as DIN SPEC 91345:2016-04 and aimed to provide a standard for the design of industrial IoT (IIoT) system [12]. It has three dimensions: product life cycle, architecture layer, hierarchy level. The architecture is targeted for industrial production, such as intelligent factories [13].

Based on reference architectures, IoT systems have been developed in academia and industry; some are counted as best practices. For example, Nijim and Ballampalli [14] proposed cloud-based architecture to control smart home system. Zeiler and Fottner [15] presented a service system for special load carriers based on RAMI4.0. As one application for the construction industry, Bottaccioli et al. [16] designed an IoT system that displayed energy usage of a building in real-time. However, the design of an IoT architecture for construction logistics remains a research gap.

Within the research project “Construction 4.0” [17], inter- views were performed with industry partners related to construction logistics. Among them were four Original Equipment Manufacturers (OEM) of the construction machines, four IoT device vendors, five construction companies, and four other service providers. As a result, the unified user requirements of a cloud-based IoT architecture are gathered following:

1) Functional Requirements:

- All IoT data from one construction project should be stored within one infrastructure.
- The architecture can deal with different data for- mats, e.g., .xml and .json.
- IoT architecture should be compatible with different communication standards, of which the most important is a machine-to-machine communication standard OPC-UA.
- There must be a multi-level access control for the data from the scope of companies, e.g., the IoT vendors can only read/write the data of their own devices from the infrastructure, whereas the construction companies can read/write all the data.

2) Non-Functional Requirements:

- Scalability
- Maintainability
- Security
- Performance
- Usability

IoT architecture for the construction logistics can be designed based on the requirements analysis. One main scope of the architecture is cloud computing. Most cloud platform providers, such as Microsoft Azure, AWS, and Google Cloud, offer infrastructure and various pre-built software services that solve some of the common problems encountered in designing scalable software architecture [18]. On the one hand, the IT infrastructures are built on top of layers of virtualization, allowing the underlying physical hardware to be shared and used by many parties simultaneously, thereby reducing the operating and ownership costs of the infrastructure. On the other hand, the software services offered by such cloud platforms solve the following design problems with the latest IT security:

- Persist and query data,
- Implement event-driven business logic,
- Stream data processing,
- Publish/subscribe based event propagation,
- Discovery, management, and communication needs of physical devices.

Despite of the similarities of the services, each cloud platform has its feature. For example, Google Cloud is featured with big data and machine learning, since its IoT core could be easily linked with BigQuery and TensorFlow [19]; Microsoft Azure provides better infrastructure for the industrial IoT (OPC-UA) applications because it has been developing OPC-UA modules since 2016 [20].

3. Architectural Design

Based on the research of the reference architectures and the requirements analysis, we have chosen the RAMI4.0 with six layers for the functional aspect. The six layers are the asset, integration, communication, information, function, and business.

However, as far as the presentation of our architecture is concerned, it is compacted down to three semantic layers, namely the edge layer, cloud layer, and business layer. Concerning the six layers mentioned above, the edge layer here encompasses layers of asset, integration, and communication. The cloud layer covers the layers of information and function. This 3-layered view of our overall architecture can be seen in Figure 1, which is referenced in the following parts.

3.1. Edge Layer

At this layer of our architecture, from the aspect of asset, we have considered three types of IoT devices: the devices from external IoT vendors to whose system we have constrained access; own IoT Edge device which we could freely configure; the IIoT devices with OPC-UA implementation. The consideration reflects the need for multiple IoT systems in construction logistics.

In terms of integration, the device software used here is required to provide three functionalities:

- Send telemetry data (push-based),
- Respond to queries,
- Execute incoming commands.

These requirements ensure that our architecture can establish a two-way data exchange between the edge and IoT application sides. A desirable consequence of having a two- way data exchange is the ability to create a closed control loop, which is the goal of creating automated IoT systems. As for the IIoT device, an OPU-UA server should be implemented and the IIoT edge modules be deployed on the device to realize the functionalities mentioned above.

From the aspect of communication, there are two main types considered. The first type is the unlocked devices that use push-based communication, which requires that the edge device be capable of directly connecting and communicating with the cloud, more precisely with the IoT service. The other type is the vendor-locked devices which involve pull-based communication. In this case, the data source, which is usually the device vendor servers, is polled regularly by time-triggered server-less applications in the cloud, allowing for the integration of both push and pull-based devices increase the compatibility with different communication standards.

The aspect of communication protocols between the edge and cloud layers is intentionally left out of the architecture, as it depends on the particular use cases and the concept of IoT device vendors. Therefore, a cloud platform with all the standard communication protocols needs to be used in the next layer.

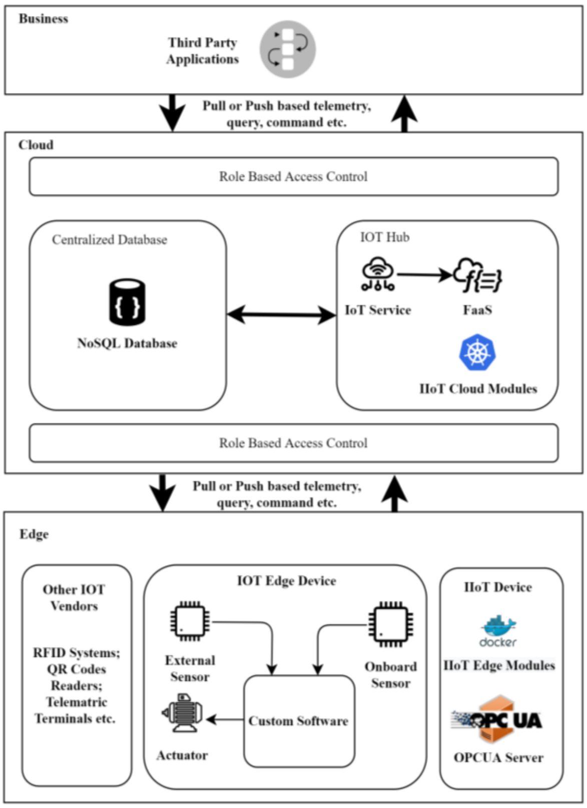


Fig. 1: Cloud-based IoT architecture for the construction industry

3.2. Cloud Layer

At the cloud layer, we have three essential services, namely No-SQL database, IoT service with FaaS, and Role-Based Access Control (RBAC). No- SQL database is meant to serve as the central repository of knowledge, storing a diverse range of telemetry data in a flexible format.

IoT service is there to connect to, monitor, and manage the IoT devices, which allows the IoT applications to consume the telemetry data and send them queries and commands. FaaS stands for function-as-a-service, and it serves to create event-driven applications in the cloud. They are also known as server-less applications. In our architecture, we propose to have two server-less applications in the FaaS. First, one of the applications deployed in the FaaS is triggered by push-based telemetry data coming into the IoT service. Then the data format and values are standardized as necessary and relayed to the database to be persisted. Afterward, another FaaS application is triggered regularly based on time to get the pull-based telemetry data, which is then sent to the database after the standardization.

For IIoT devices, a similar functionality is achieved by having the IIoT cloud modules which allows us to present a central interface to the IoT applications for interacting with the IIoT devices.

There is a secure access mechanism called role-based access control on both sides of the cloud layer facing the other layers. It provides the ability to grant fine-grained access to the cloud resources. Therefore, the requirement of having a multi-level access control is satisfied.

These cloud services are scalable and maintainable due to the virtualized and tenancy-based nature of cloud resources. In addition, usability and security are granted due to the high documentation standard.

3.3. Business Layer

Finally, at the consumer end, an IoT application can access both push-based and pull-based telemetry data from the central No-SQL database. The IoT application can also send queries and commands to push-based edge devices through the IoT service. This ability to send queries and commands enables the application to execute its business logic given the telemetry data.

Generally, this simple architecture fulfils the requirements in three aspects:

- Separation of concerns between data sources and sinks, due to the central database and standardized data format,
- Cloud-to-machine, machine-to-cloud, and machine-to- machine communication through the IoT service,
- Secure and scalable platform for the storage and consumption of telemetry data.

4. Implementation and Validation

In the design of our architecture, two types of use cases were identified and considered:

- Vanilla case: Concerns edge devices and IoT applications.
- OPC-UA case: Concerns edge devices and IoT applications that use OPC-UA structure.

This paper implemented the previously proposed architecture within the Microsoft Azure infrastructure and validated it with the two use cases.

4.1. Vanilla Case

Figure 2 shows how the current implementation looks concerning the architecture in1. The push-based and pull-based devices are distinguished. Because pull-based devices cannot communicate with the IoT service for various reasons such as vendor-locked firmware, whereas push-based ones can directly communicate.

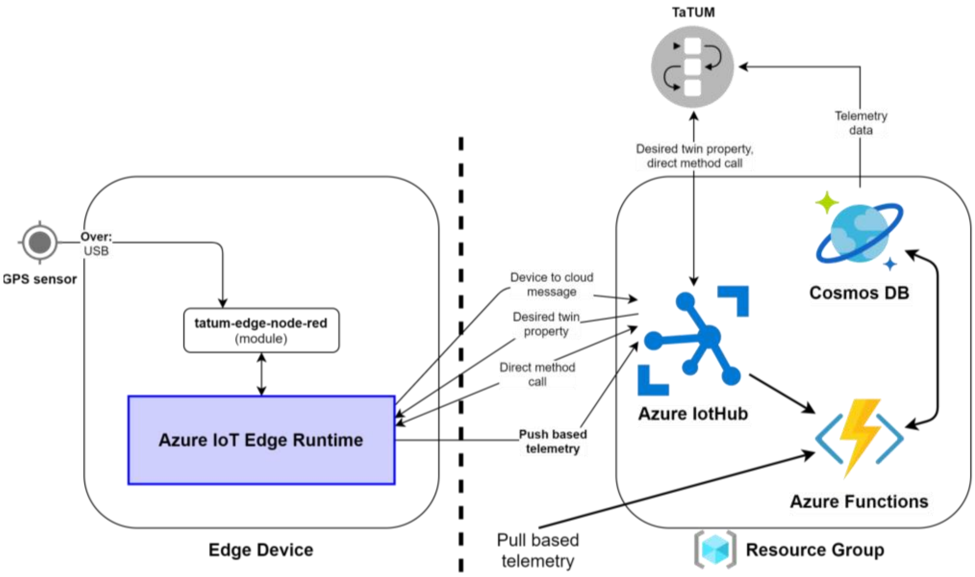


Fig. 2: Implementation with IoT Edge device - Vanilla case

Edge Layer: The custom device software is realized by Azure IoT Edge Run- time, and the custom module, named tatum-edge-node-red (Figure3), is developed to run in this Runtime environment. In addition, Azure IoT Edge Runtime allows automatic management of the life cycle of containerized applications in a somewhat similar way to Kubernetes. These containerized applications are called modules in Azure terminology [21].

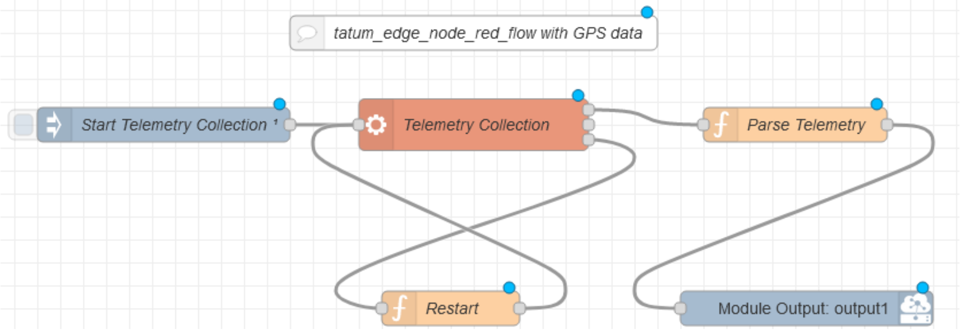
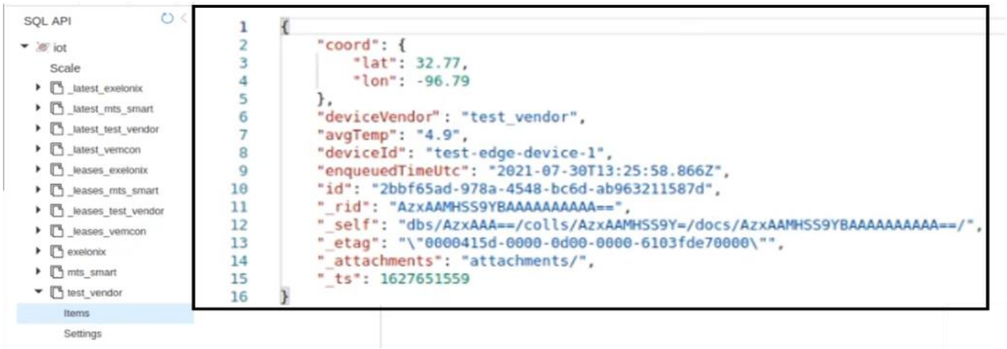

tatum_edge_node_red_flow with GPS data
Start Telemetry Collection ¹
Telemetry Collection
Parse Telemetry
Restart
Module Output: output1

Fig. 3: Tatum-edge-node-red flow [22]

The name of our module stems from the IoT application for which the architecture was designed, named TaTUM, and the fact that this module was implemented using node-red. It allows sending push-based telemetry, sending device- to-cloud messages, receiving desired twin properties, and receiving direct method calls like remote procedure calls (RPC). In terms of the sensing and acting capabilities, we used a GPS sensor that was regularly polled by the custom module software, and the data was pushed onto the IoT service.

Cloud Layer: The conceptual cloud services were replaced by the ones provided by Azure, namely Azure Cosmos DB, IotHub, and Functions. Their functionality was the same as described in chapter III. The push-based data was sent to Azure IotHub by the IoT devices that were directly connected. We had time-triggered server-less applications running in Azure Functions and polling the data sources for the pull-based data. Again, using Azure Functions, we had applications triggered by the arrival of telemetry data into Azure IotHub. Through these FaaS applications, all telemetry data was fed into Cosmos DB after standardizing the data format.



```
{
    "coord": {
        "lat": 32.77,
        "lon": -96.79
    },
    "deviceVendor": "test_vendor",
    "avgTemp": "4.9",
    "deviceId": "test-edge-device-1",
    "enqueuedTimeUtc": "2021-07-30T13:25:58.866Z",
    "id": "2bbf65ad-978a-4548-bc6d-ab963211587d",
    "_rid": "AzxAAMHSS9YBAAAAAAAAAA==",
    "_self": "dbs/AzxAAA==/colls/AzxAAMHSS9Y=/docs/AzxAAMHSS9YBAAAAAAAAAA==/",
    "_etag": "\"0000415d-0000-0d00-0000-6103fde70000\"",
    "_attachments": "attachments/",
    "_ts": 1627651559
}
```

Fig. 4: Data collection in the centralized CosmosDB.

Cosmos DB filled in the role of a No-SQL database as a scalable solution with a REST API interface. Each telemetry data was stored in a JSON format with a few standard fields. These fields were message ID, device vendor, device ID, and UTC timestamp in our implementation. The message ID field was mandatory to index and partition the messages. The device vendor field was there to identify the device vendor from which the message originated and find the whole message format, which was fixed per device vendor. Also, all these data fields allowed to write complex message filters. The collected data from all vendors are shown in Figure 4.

Business Layer: We implemented a containerized and Kubernetes-ready application called TaTUM for this layer. It is designed for managing a fleet of construction devices from a web-based interface.

4.2. OPC-UA Case

In the case of OPC-UA, an open-source project called Azure Industrial IoT (IIoT) was utilized to integrate the OPC-UA standard's capabilities into our overall architecture. Its conceptual diagram can be seen in Figure 5, which is also implicitly part of the overall architecture in Figure 1.

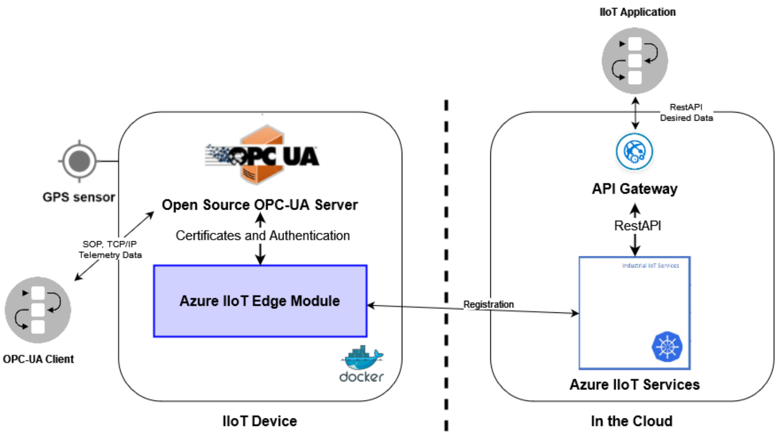
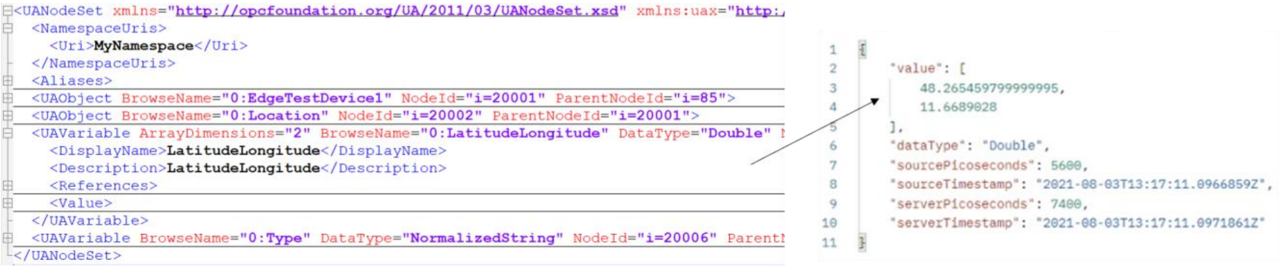


Fig. 5: Implementation with IIoT devices - OPC-UA case

Edge Layer: On left side of Figure 5, as depicted there, we had a custom developed OPC-UA server running in an edge device. Besides, Azure IIoT edge modules also ran on the same edge device in our implementation. These edge modules were responsible for discovering, monitoring, and interacting with OPC-UA servers. They bridged the connection gap between OPC-UA servers and IIoT edge modules. Due to their containerized design, the edge modules were able to run in Azure IoT edge runtime. The edge modules were deployed in the runtime to ensure the automatic management of their life cycle.

As in the vanilla case, we have a GPS sensor in terms of sensing and acting capabilities. Our OPC-UA server [23] responds to incoming requests for the current location information from the sensor. The OPC-UA information structure used in our server implementation is shown in Figure 6.



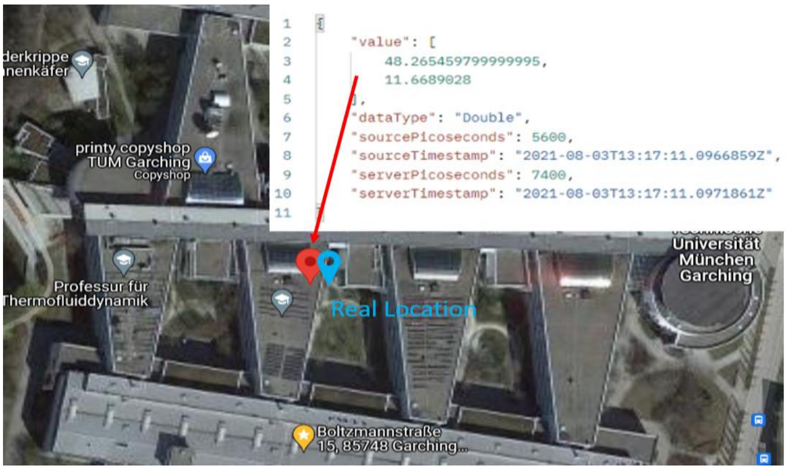
```
<UANodeSet xmlns="http://opcfoundation.org/UA/2011/03/UANodeSet.xsd" xmlns:uax="http:/
  <NamespaceUris>
    <Uri>MyNamespace</Uri>
  </NamespaceUris>
  <Aliases>
  <UAObject BrowseName="0:EdgeTestDevice1" NodeId="i=20001" ParentNodeId="i=85">
  <UAObject BrowseName="0:Location" NodeId="i=20002" ParentNodeId="i=20001">
  <UAVariable ArrayDimensions="2" BrowseName="0:LatitudeLongitude" DataType="Double" N
    <DisplayName>LatitudeLongitude</DisplayName>
    <Description>LatitudeLongitude</Description>
    <References>
    <Value>
  </UAVariable>
  <UAVariable BrowseName="0:Type" DataType="NormalizedString" NodeId="i=20006" ParentN
</UANodeSet>
```

```
{
    "value": [
        48.265459799999995,
        11.6689028
    ],
    "dataType": "Double",
    "sourcePicoseconds": 5600,
    "sourceTimestamp": "2021-08-03T13:17:11.0966859Z",
    "serverPicoseconds": 7400,
    "serverTimestamp": "2021-08-03T13:17:11.0971861Z"
}
```

Fig. 6: OPC-UA information structure

Cloud Layer: As in the right half of Figure 5, Azure IIoT cloud modules were deployed into a Kubernetes cluster. Due to its containerized design, IIoT cloud modules supported running on a Kubernetes cluster, which brought the advantage of scalability. These cloud modules were responsible for registering, monitoring, and interfacing with the edge modules. So, an IoT application can interact with the OPC-UA enabled edge devices via the API gateway of the IIoT cloud modules and use the functionality of OPC- UA such as data access (DA), alarms & conditions (A&C), and historical access (HA) [24].

Business Layer: To demonstrate the viability of the used solution, Azure IIoT, we had a collection of REST API query templates for use in Postman. The API Gateway was tested successfully by requesting GPS data and displaying it on Google Maps (Figure7).



```
{
    "value": [
        48.265459799999995,
        11.6689028
    ],
    "dataType": "Double",
    "sourcePicoseconds": 5600,
    "sourceTimestamp": "2021-08-03T13:17:11.0966859Z",
    "serverPicoseconds": 7400,
    "serverTimestamp": "2021-08-03T13:17:11.0971861Z"
}
```

Fig. 7: GPS data requested through OPC-UA API Gateway

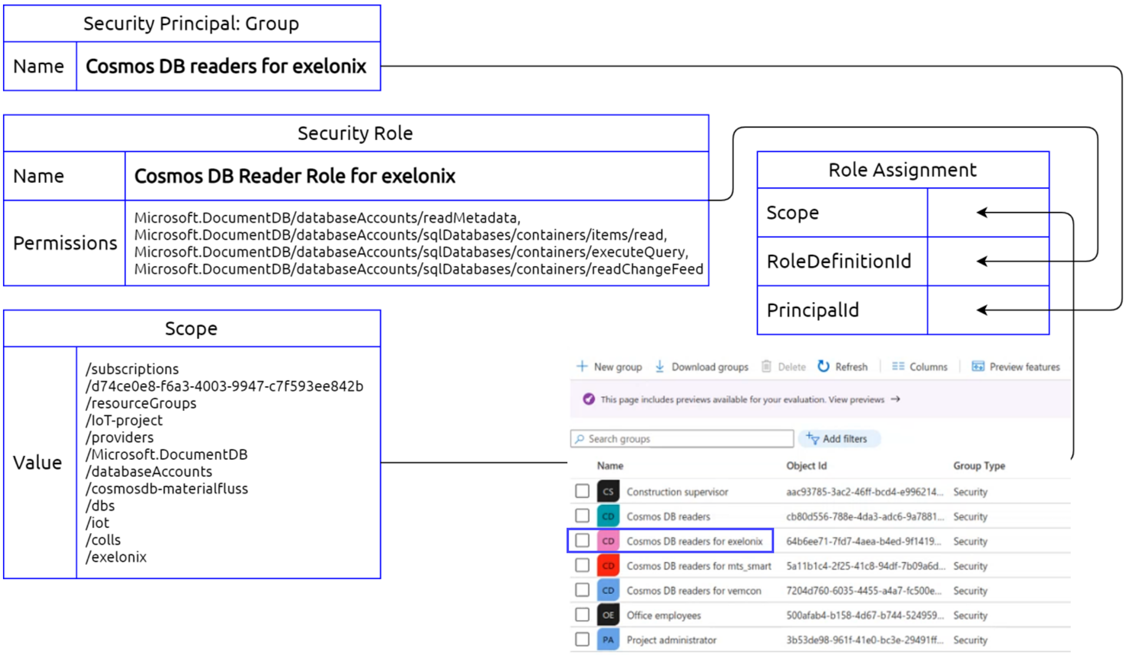
4.3. RBAC

Role-based access control was implemented using Azure RBAC. It was mainly utilized to set up access into the database table of various device vendors, such as MTS Smart, Vemcon, and Exelonix. There are three fundamental components in Azure RBAC that enable this.

- Security principal: A digital identity which has an associated way of proving its identity using authentication methods. In Azure, this authentication is provided using a secure key or an X509 certificate.
- Role definition: A list of permissions, each defined by a string that tells what kind of operation is permitted.

- Scope: Name of the resource level at which the role definition applies.

The three components together comprise a role assignment. A **role assignment** refers to a security principal, a role definition, and a scope. In this way, it allows the security principal to use the permissions defined in the role definition within the specified scope. The concrete implementation, which can be seen in Figure 8, was based on the requirements analysis results.



Security Principal: Group	
Name	**Cosmos DB readers for exelonix**

Security Role	
Name	**Cosmos DB Reader Role for exelonix**
Permissions	Microsoft.DocumentDB/databaseAccounts/readMetadata, Microsoft.DocumentDB/databaseAccounts/sqlDatabases/containers/items/read, Microsoft.DocumentDB/databaseAccounts/sqlDatabases/containers/executeQuery, Microsoft.DocumentDB/databaseAccounts/sqlDatabases/containers/readChangeFeed

Role Assignment	
Scope	
RoleDefinitionId	
PrincipalId	

Scope	
Value	/subscriptions /d74ce0e8-f6a3-4003-9947-c7f593ee842b /resourceGroups /IoT-project /providers /Microsoft.DocumentDB /databaseAccounts /cosmosdb-materialfluss /dbs /iot /colls /exelonix

This page includes previews available for your evaluation. View previews →

		Name	Object Id	Group Type
□	CS	Construction supervisor	aac93785-3ac2-46ff-bcd4-e996214...	Security
□	CD	Cosmos DB readers	cb80d556-788e-4da3-adc6-9a7881...	Security
□	CD	Cosmos DB readers for exelonix	64b6ee71-7fd7-4aea-b4ed-9f1419...	Security
□	CD	Cosmos DB readers for mts_smart	5a11b1c4-2f25-41c8-94df-7b09a6d...	Security
□	CD	Cosmos DB readers for vemcon	7204d760-6035-4455-a4a7-fc500e...	Security
□	OE	Office employees	500afab4-b158-4d67-b744-524959...	Security
□	PA	Project administrator	3b53de98-961f-41e0-bc3e-29491ff...	Security

Fig. 8: Implementation of RBAC based on requirements analysis

5. Conclusion

This paper presents an IoT architecture for the construction industry, which includes the use of different IoT systems and the OPC-UA standard to meet industrial applications' needs. Another advantage of this architecture is that the construction industry's complex hierarchy of information systems is conceived. With the implementation of two use cases, Vanilla and OPC-UA, the usefulness of this architecture is demonstrated. This architecture enables the usage of various IoT technologies to suit the needs of construction logistics within the same infrastructure and ensure the data consistency, security, and flexibility. As a result, the efficiency of the working processes in construction logistics would be greatly improved.

In the future, more sensors can be combined with edge devices at the edge layer and more applications can be developed in the cloud to enrich the use cases in real-world scenarios.

6. Acknowledgements

This research was funded by the German Federal Ministry of Education and Research within the research project Building 4.0. We kindly thank our industry partner for providing the data.

7. References

[1] R. Agarwal, S. Chandrasekaran, and M. Sridhar. *Imagining construction's digital future*. 2016. McKinsey. [Online]. Available: https://www.mckinsey.com/business-functions/operations/our insights/imagining-constructions- digital-future

[2] Statistisches Bundesamt. *Volkswirtschaftliche Gesamtrechnungen*. 2020, Table 2.13.

[3] McKinsey Global Institute. *Reinventing construction: A route to higher productivity*. McKinsey & Company, February 2017.

[4] E. O. Ibem and S. Laryea. *Survey of digital technologies in procurement of construction projects*. vol. 46, pp. 11–21, 2014, pII: S092658051400154X.

[5] International Telecommunication Union. *Series y: Global information infrastructure, internet protocol aspects and next-generation networks*. International Telecommunication Union, 2012.

[6] S. Dilakshan, A. P. Rathnasinghe, and L. I. P. Seneviratne. Potential of internet of things (iot) in the construction industry. In *Proceedings of the 9th World Construction Symposium 2021 on Reshaping construction: Strategic, Structural and Cultural Transformations towards the 'Next Normal'.* The Ceylon Institute of Builders Sri Lanka, 2021, pp. 445–457.

[7] W. A. Günthner, E. Rank, N. Vogt, T. Euringer, W. Stockbauer, E. Hartmann, and G. Hirzinger. *Forbau-virtuelle Baustelle - digitale Werkzeuge für die Bauplanung und – Abwicklung*. 2010.

[8] J. P. van Leeuwen and S. Fridqvist. *An information model for collaboration in the construction industry.* vol. 57, no. 8-9, pp. 809–816, 2006, pII: S0166361506000881.

[9] M. U. Farooq, M. Waseem, S. Mazhar, A. Khairi, and T. Ka- mal. *A review on internet of things (iot)*. vol. 113, no. 1, pp. 1–7, 2015.

[10] A. Bassi, M. Bauer, M. Fiedler, T. Kramp, R. van Kranenburg, S. Lange, and S. Meissner. *Enabling Things to Talk*. Springer Berlin Heidelberg. 2013.

[11] V. Tsiatsis, S. Karnouskos, J. Höller, D. Boyle, and C. Mulligan. *Internet of things*. Second edition. Academic Press. 2019.

[12] Din Spec 91345:2016-04, Referenzarchitekturmodell Industrie 4.0 (rami4.0). Beuth Verlag GmbH.

[13] A. Rojko. *Industry 4.0 concept: Background and overview*. International Journal of Interactive Mobile Technologies, vol. 11, no. 5, 2017.

[14] D. B. Mais Nijim. The design of a novel smart home control system using a smart grid based on edge and cloud computing. *International Journal of Smart Grid and Clean Energy*, no. 11, pp. 57–11, 2022.

[15] J. Zeiler and J. Fottner. Architectural design for special load carriers as smart objects in a cloud-based service system. in *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*. IEEE, 42019, pp. 644–652.

[16] L. Bottaccioli, A. Aliberti, F. Ugliotti, E. Patti, A. Osello, E. Macii, and A. Acquaviva. Building energy modelling and monitoring by integration of iot devices and building information models. in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*. IEEE, 72017, pp. 914–922.

[17] Technical University Munich, Technical University Dresden. *Bauen 4.0 – Effizienz und Produktivitätssteigerung von Bauprozessen durch Vernetzung und Kommunikation mobiler Maschinen.* [Online]. Available: https://www.plattformi40.de/IP/Redaktion/DE/Anwendungsbeispiele/525-TUDresden/BeitragTU Dresden.html. 2019.

[18] O. Alzakholi, L. Haji, H. Shukur, R. Zebari, S. Abas, and M. Sadeeq. *Comparison among cloud technologies and cloud performance*. Vol. 1, no. 2, pp. 40–47, 2020.

[19] Google Cloud. *Iot core*. [Online]. Available: https://cloud.google.com/iot-core/. 2022.

[20] OPC Foundation. *Microsoft and opc foundation demonstrate "azure industrial iot" with 40 opcua enabled walls in their international technology centers (mtc)*. 2017.

[21] "What is azure iot edge," Sep 2021. [Online]. Available: https://docs.microsoft.com/en-us/azure/iot-edge/about-iotedge?view=iotedge-2020-11

[22] "tatum-edge-node-red", 2021. [Online]. Available: https://github.com/denizetkar/tatum-edge-node-red

[23] "opcua edge server", 2021. [Online]. Available: https://github.com/denizetkar/opc-ua-edge-server

[24] "Opc unified architecture specification." [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture