Generation of hierarchical OPC UA-Servers from AutomationML-Models

Michael Okon
Information Management and Production Control (ILT)
Fraunhofer Institute of Optronis, System Technologies and Image Exploitation IOSB
Karlsruhe, Germany
michael.okon@iosb.fraunhofer.de

Ljiljana Stojanovic
Information Management and Production Control (ILT)
Fraunhofer Institute of Optronis, System Technologies and Image Exploitation IOSB
Karlsruhe, Germany
ljiljana.stojanovic@iosb.fraunhofer.de

***Abstract*— Fraunhofer IOSB has developed an AML2UAConverter that transforms an AutomationML model in an UA-XML-Nodeset. This has been done based on the rules of the DIN SPEC 16592 – Combining OPC Unified Architecture and Automation Markup Language (published in December 2016), which is built on the OPC UA companion specification „AutomationML for OPC UA“ (Version February 2016).**

Furthermore, an expanded AutomationML-File, which describes the new OPC UA (Aggregation)-Server properties and its DataVariables (Version May 2017), is provided:

- **The new server is described with its properties, such as DiscoveryURL, NameSpaceTable and more.**
- **For each DataVariable-Attribute in the initial model an attribute of type OPC UA-DataVariable is created for the new server.**

The converted UA-XML-Nodeset is used for automatic creation of C-Code for an aggregating OPC UA-Server based on the open62541 toolkit. For all DataVariables of type OPC UA available in the initial AutomationML-File an integrated OPC UA Client, which subscribes to all variables in the underlying OPC UA-Servers, is integrated into the aggregating server automatically.

In the project iTEXFer an enhanced multilevel concept for gener**ating OPC UA-Servers and a Master OPC UA-Server from AutomationML Models has been developed. The steps are:**

1. **The AutomationML Models that model different aspects (e.g. different stations in a production line) are combined in an AutomationML Container.**
2. **This container is used for the generation of several UA-XML NodeSets (one for each AutomationML model) and expanded AutomationML-Files.**
3. **Combining these generated AutomationML files into one master AutomationML file is the basis for the next generation step: The Master-UA-XML NodeSet and the master AutomationML file for the corresponding MasterAggregation Server, which references only the servers generated before.**
4. **All the converted UA-XML NodeSets can again be used for creating C-Code for aggregating OPC UA-Servers based on the open62541 toolkit. So, the Master OPC UA-Server of step 3 represents a hierarchical OPC UA Server referencing the OPC UA-Servers of step 2.**

The proposed concept can be used for the generation of hierarchical OPC UA Servers with arbitrary layers – simply by applying it in an iterative way. The approach is evaluated in the context of the iTEXFer project by describing the STFI's demo production line.

Some benefits of this concept are:

- **Automatic generation of OPC UA-Nodesets from AutomationML-Files considering hierarchical aspects**
- **Dynamic modifications of OPC UA-Nodesets after AutomationML model changes**
- **Consistency of data models because of automatic generation and modification**
- **OPC UA-Wrapping of (not only OPC UA) variables via the DataVariable concept**

In future the concept could be enhanced for integrating Industrie 4.0 concepts as „Asset Administration Shell“ Submodels in AutomationML and UA-XML NodeSets for hierarchical layer views.

Keywords—AutomationML, OPC UA, UA-XML-Nodeset, DIN SPEC 16592, open62541, iTEXFer

I. INTRODUCTION

The following explanations refer to the CAEX [1] version 2.15 on which AutomationML is based. Extensions and changes in CAEX version 3.0 are mentioned explicitly.The DIN specification DIN SPEC 16592 - Combining OPC Unified Architecture and Automation Markup Language (December 2016) [2], which is based on the OPC UA Companion Specification AutomationML for OPC UA (version February 2016) [3], describes how to transform an AML model into an OPC UA model using special mapping rules. An automatic generation of an OPC UA server based on open52641 [4] using the AML2UACenter [5] is possible.

The applied concepts are also described in more detail in [6] and [7].

The main mapping rules for AML elements on OPC UA elements can be found in Fig. 1 and Fig. 2. For example, an AML InternalElement is mapped to an OPC UA Object and an AML Attribute to an OPC UA Variable.[1]

[1] OPC UA Objects and OPCUA Variables are standardized entities in an OPC UA address space.

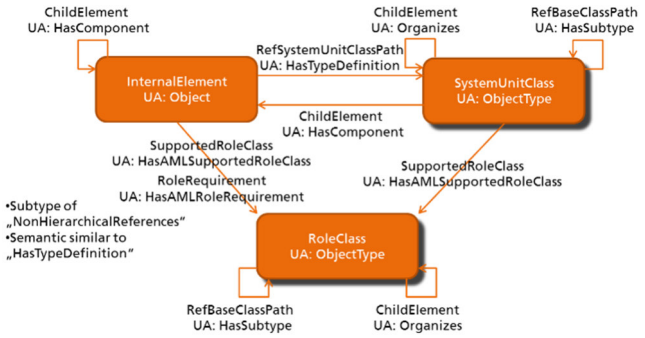

Fig. 1. AML2UA mapping, base elements

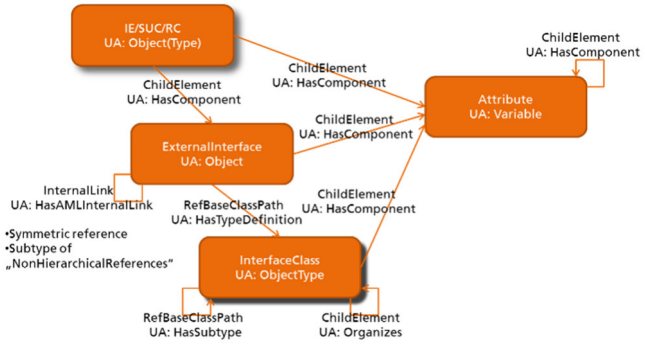

Fig. 2. AML2UA mapping, basic elements

An extension to an automatic generation of hierarchical OPC UA servers, which result from several AutomationML descriptions, has been developed in the iTEXFer [8] project and integrated into the existing AML2UACenter.

II. The Concept and its Implementation

A. The current AML2UAConverter-Capabilities

Only a few components are needed for the generation of simple aggregating OPC UA servers:

AML2UAConverter

In the engineering phase, the AML2UAConverter component converts an AutomationML file into an OPC UA NodeSet[2] according to the "OPC UA Information Model for AutomationML" or "DIN SPEC 165922016-12" specification and an extended AutomationML file (for aggregating server). It interacts with other components within the AML2UACenter, but can be considered as standalone and independent. (see Fig. 3).

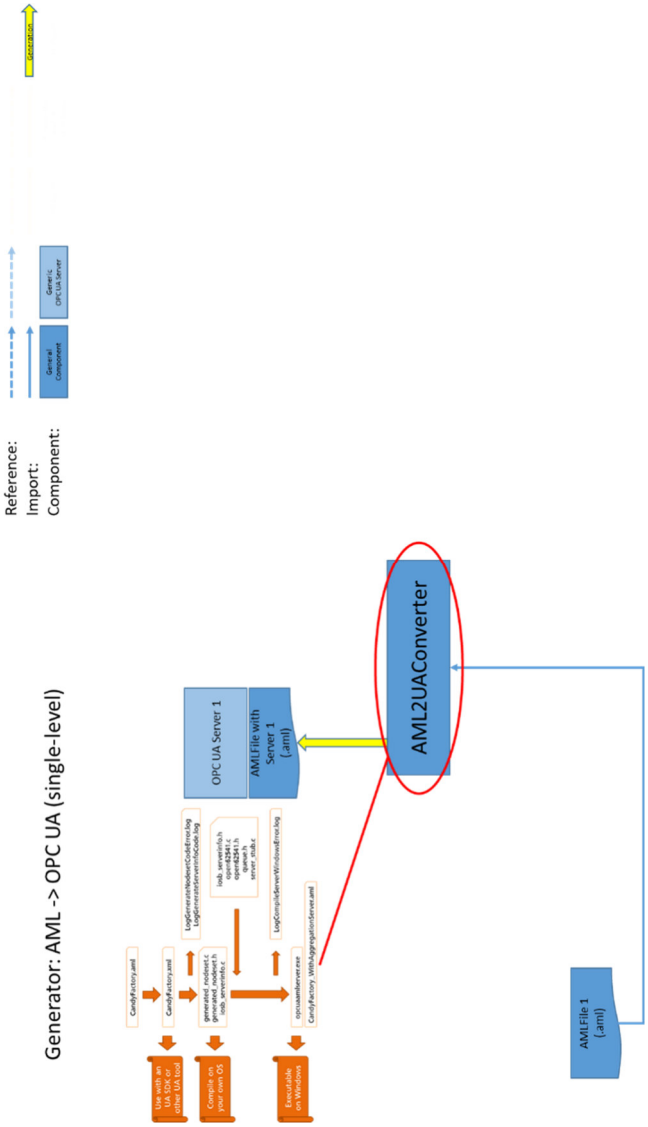

Fig. 3. The current AML2UAConverter, embedded in the AML2UACenter

AML2UACenter

The AML2UACenter component is a framework for generating OPC UA servers, node sets and extended descriptions (for aggregating servers) from individual AML files. The main component is the AML2UAConverter described above. The underlying platform is WebGenesis® [9] and is hosted by an Apache Tomcat. In addition to the output of the AML2UAConverter, it provides source code (in the programming language C) based on open62541 and an executable OPC UA server via an integrated compiler/linker for Windows 10. The start page is shown in Fig. 4. After registration and subsequent personal login, a user can upload his or her AutomationML-compliant files, start the generation and download the described output.

[2] An OPC UA Nodeset is a standardized XML description of the information model of an OPC UA server.

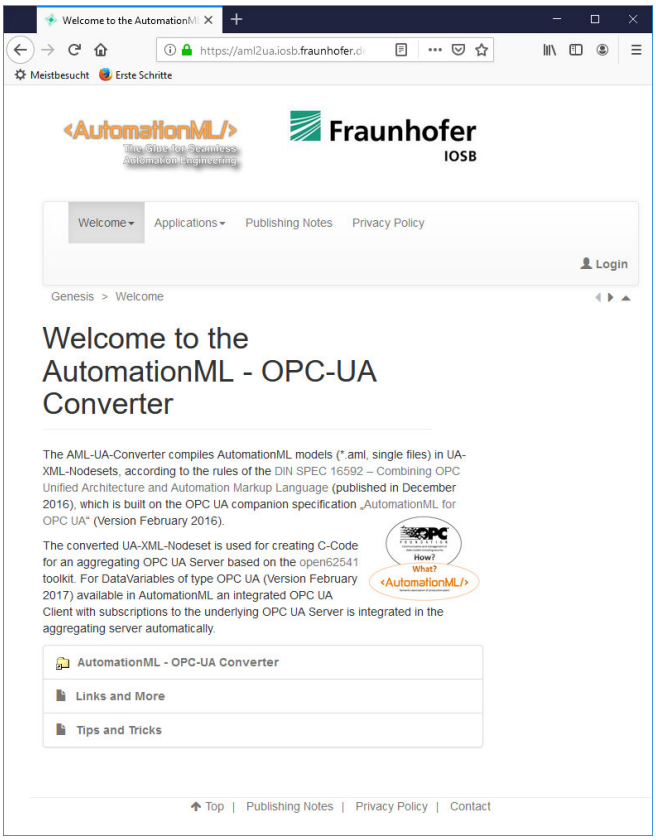

Fig. 4. The IOSB AML2UACenter

B. The extended AML2UAConverter-Capabilities

The existing concept was extended to generate hierarchical OPC UA servers: new components had to be added and existing ones extended:

AML2UAConverter

The component AML2UAConverter did not need to be changed.

AMLConfigurator

In the engineering phase, the AMLConfigurator component creates an AutomationML container (AMLX) from the given AutomationML files (for the description of individual plants - with process connection via the DataVariable concept). The component does not interact with any other component. It can be regarded as independent. Fig. 5 shows the user interface of the component.

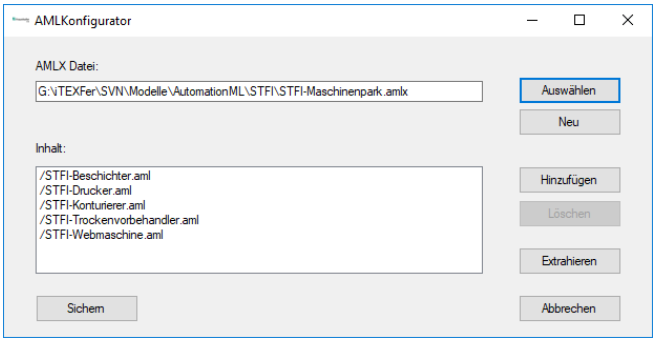

Fig. 5. The IOSB AMLConfigurator

Note: The AutomationML Editor of the AutomationML Consortium provides similar functionality.

AML2UAAggregator

In the engineering phase, the AML2UAAggregator component creates the associated OPC UA NodeSets and extended AutomationML descriptions (with additional descriptions of the aggregated servers) from the AutomationML files contained in an AutomationML container (for the description of individual plants with process connection according to the DataVariable concept) and aggregates these into an integrated description of the plant in an AutomationML file (master). It interacts with other components within the AML2UACenter, but can be regarded as independent.

AML2UACenter

The AML2UACenter component was enhanced with the components described above; its interactions are shown in Fig. 6. The user interface presents itself to the user in the same form as before, the generation logic is recognized by the input: AML files are treated as before, AMLX files are treated according to the new logic.

Interactions

The AML2UAConfigurator component groups all the underlying AML files together in an AutomationML container, which is imported by the AML2UAAggregator and processed into a new master AML file. The "intermediate products" generated by the AML2UAConverter (NodeSets and C code) can be used to generate executable OPC UA servers. This iterative procedure was transparently integrated into the AML2UACenter. Using the DataVariable concept, the newly created master component (as a hierarchical OPC UA server) is directly linked to the subordinate servers. In a further step, this master file can be prepared by means of an editor with entries for the executability of the servers (a posteriori Discovery URL [3] Configuration). The prepared master can then be processed by the AML2UAConverter in the old way. The complete process of generating a hierarchy level with the resulting "intermediate products" is shown in Fig. 6. The iterative application of this procedure then delivers an arbitrary number of hierarchy levels.

[3] The DiscoveryURL describes a connection to an OPC UA Server in a network.

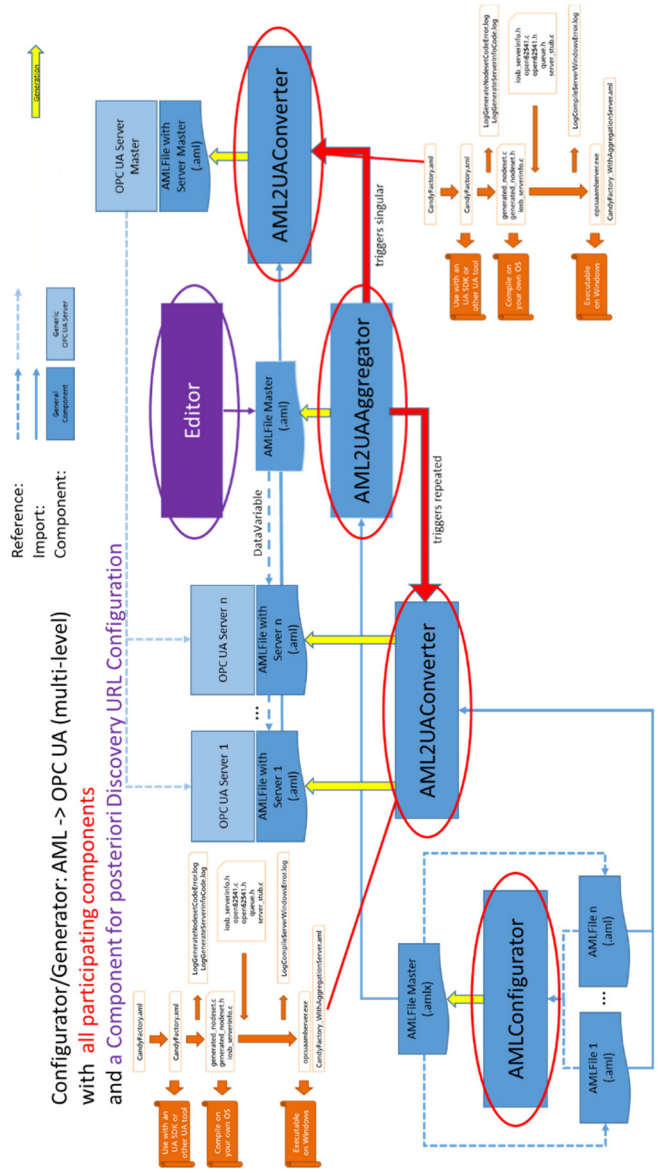

Fig. 6. Interactions between the components, partly embedded in the new AML2UACenter

III. Workflows for User

For a user of the above concept, the following workflows result. They refer to the initial conversion and post-conversion, whereby it is important to ensure that all AML files considered comply with the AML specifications and rules:

A. Workflow for Initial Conversion

1. Create AML files.
2. Check created files for correctness in the AMLTest-Center and correct them if necessary.
3. Integrate files into an AML container.
4. Make AML containers available in the AML2UACenter for generation and generate NodeSets, code and executable servers, as well as the master AML file.
5. Carry out any subsequent changes to the master AML file on your own responsibility, in particular to the DiscoveryURL, test it for correctness in the AMLTest Center and correct it if necessary.
6. Provide the (modified) AML master file in the AML2UACenter for generation and generate NodeSet, code and executable server.
    a. Online-modification:
    Import changed node sets and, if necessary, reset subscriptions by the servers involved.
    b. Offline-modification:
    Recompile generated code and link, then start server, or start finished generated server. Any existing subscriptions of external clients must be set up again.

B. Workflow for post-conversion (changes in one or more AML files)

1. Modify the file(s) ensuring that elements describing the same entity retain their AutomationML ID[4].
2. Check changed file(s) for correctness in the AMLTestCenter and correct if necessary.
3. Replace the file(s) in the AML container with the modified file(s).
4. Provide the AML container in the AML2UACenter for generation and generate NodeSets, code and executable servers, as well as master AML file.
5. Carry out any subsequent changes to the master AML file on your own responsibility, in particular to the DiscoveryURL, test it for correctness in the AMLTestCenter and correct it if necessary.
6. Use the (modified) AML master file in the AML2UACenter for generation of NodeSet, code and executable server.
    a. Online-modification:
    Import changed node sets and, if necessary, have subscriptions reset by the servers involved.
    b. Offline-mofification:
    Recompile generated code and link, then restart server, or restart finished generated server. Any existing subscriptions of external clients must be reset.

Fig. 7 combines the two workflows in one representation.

[4] The AML2UAConverter uses the AutomationML ID to generate an unique NodeId in the NodeSet of an OPC UA server

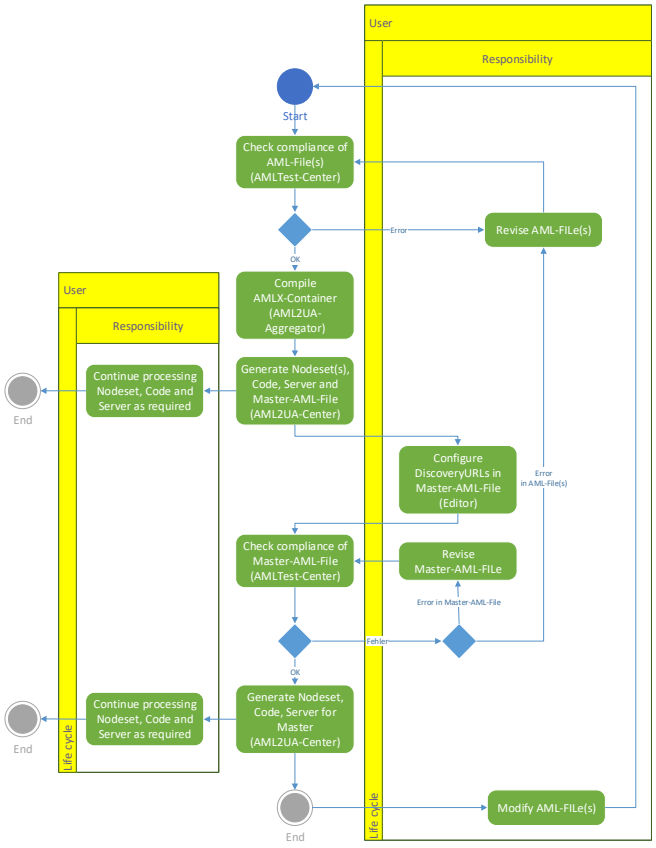

Fig. 7. Workflows for users

IV. Example

The iTEXFer project addresses, in one of its sub-areas, the "uniform" representation - using AutomationML - of the components involved in a complete production line in an area of the textile industry. Detailed knowledge - e.g. of the process flow - is not required for the further discussion, since only the principle of the concept described above is to be clarified. Likewise, modeling in the AutomationML files of the participating components in English was dispensed with.

The production line consists of several processing and transport components, which are first "hierarchized" into a machine and transport park and then into an overall view. The occurring combination "STFI" in the designations is due to the fact that this demonstration line is set up at the **S**ächsisches **T**extil **F**orschungs **I**nstitut (Saxon Textile Research Institute) in Chemnitz (see also Fig. 5 for the hierarchy level "Machinery").

Fig. 8 shows the desired final configuration, in which there is currently only one subordinate OPC UA server from B&R as a process-oriented external component for the weaving machine, which is already connected using the DataVariable concept.

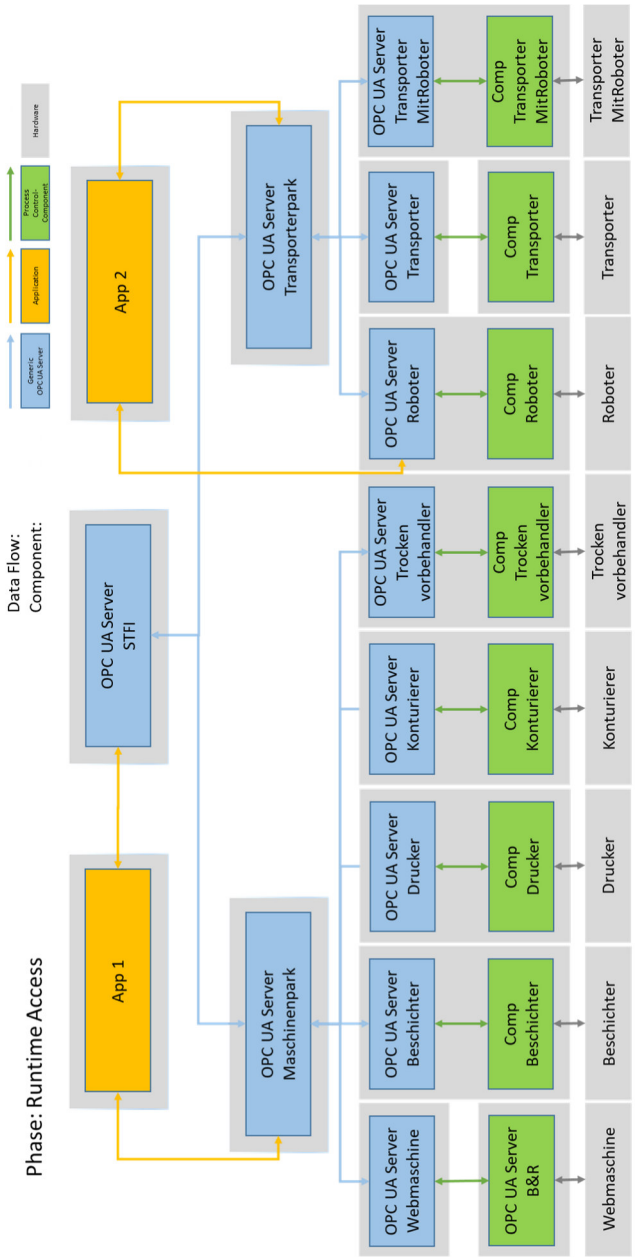

Fig. 8. The desired hierarchical structured servers

For this purpose, the "uniformly" created AML descriptions

- STFI-Beschichter (coater)
- STFI-Drucker (printer)
- STFI-Konturierer (contouring tool)
- STFI-Trockenvorbehandler (dry pretreater)
- STFI-Webmaschine (weaving machine)

were firstly combined to STFI-Maschinenpark (machinery) and the AML descriptions

- STFI-Roboter (robot)
- STFI-Transporter (transporter)
- STFI-TransporterMitRoboter (transporter with robot)

were combined to form the STFI Transporter Park.

The generated AML descriptions

- STFI-Maschinenpark (machinery)
- STFI-Transporterpark (transport fleet)

were finally combined to form the global description STFI.

The iterative generation provided a total of 11 executable OPC UA servers in 3 generated hierarchy levels. The external OPC UA server as a process-oriented component thus forms the 4th level. The other process-related components are to be integrated analogously later.

The generated OPC UA servers can all be addressed separately and present themselves in an OPC UA client as in **Fehler! Verweisquelle konnte nicht gefunden werden.**. Value changes are propagated to each hierarchy level.

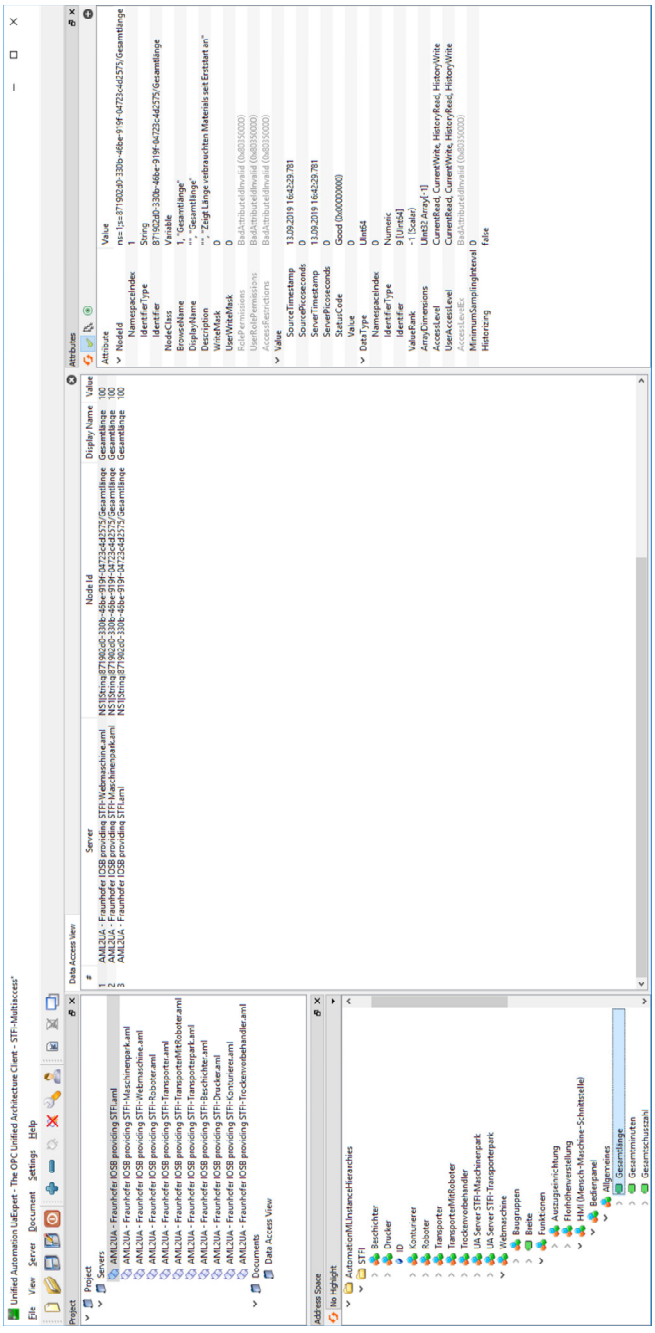

Fig. 9. The hierarchical servers in an OPC UA client

V. Summary and Outlook

This paper describes the extension of the existing concept for generating an aggregating OPC UA server from an AutomationML file to the concept for generating hierarchical OPC UA servers from an AutomationML container. The multiple iterative application of this concept leads to a hierarchical arrangement of OPC UA servers.

Workflows for the use of the available implementations of this concept were shown, in particular for changes in the underlying AutomationML files with

Benefits of this concept are as follows:

- Automatic generation of OPC UA-Nodesets from AutomationML-Files considering hierarchical aspects
- Dynamic modifications of OPC UA-Nodesets after AutomationML model changes
- Consistency of data models because of automatic generation and modification
- OPC UA-Wrapping of (not only OPC UA) variables via the DataVariable concept

The concept itself was evaluated with an example using a multi-level hierarchy in the iTEXFer project.

The following enhancements to the concept in the Asset Administration Shell (AAS) of Plattform Industrie 4.0 [10] seem possible:

- Conversion of the mapping of the AAS in AutomationML to OPC UA
- Creation of submodels as OPC UA server views or independent servers

In both cases, however, the results of the current activities on the mapping of the AAS specifications on to AutomationML and OPC UA need to be completed by Plattform Industrie 4.0.

Acknowledgment

The concept presented was developed within the framework of the iTEXFer project.

References

[1] https://www.plt.rwth-aachen.de/cms/PLT/Forschung/Projekte2/~ejwy/CAEX-IEC-62424/

[2] (2016). DIN Spec 16529: Combining OPC Unified Architecture and Automation Markup Language. Beuth.

[3] https://opcfoundation.org/developer-tools/specifications-unified-architecture/opc-unified-architecture-for-automationml/

[4] https://open62541.org/

[5] https://aml2ua.iosb.fraunhofer.de/

[6] R. Henssen, M. Schleipen. (2014). Interoperability between OPC-UA and AutomationML. Disruptive Innovation in Manufacturing Engineering towards the 4th Industrial Revolution, Proceedings of the 8th International CIRP Conference on Digital Enterprise Technology - DET. Stuttgart: Fraunhofer Verlag.

[7] https://www.automationml.org/o.red.c/news-173.html

[8] http://www.stfi.de/forschungsvorhaben/umsetzungsvorhaben/itexfer

[9] https://www.iosb.fraunhofer.de/servlet/is/18052/

[10] https://www.plattform-i40.de/