23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

Implementing Industry 4.0 Asset Administrative Shells in Mini Factories

Alejandro Seif\textsuperscript{a}, Carlos Toro\textsuperscript{a}, Humza Akhtar\textsuperscript{a}

\textsuperscript{a}Advanced Remanufacturing and Technology Centre (ARTC) – Agency for Science, Technology and Research (A*STAR), (Singapore)

Abstract

Physical assets in manufacturing, can be: machines, sensors, actuators and in general everything that can be part of the plant and its processes. In such domain, the search for “agility” has come to be one of the most requested features. As a result, considerable research has been conducted to design manufacturing systems which embody manufacturing flexibility. It is in this context where the evolution of the supply-chain based concept of Mini Factories is relevant, as it allows a network-like interconnectivity between assets by setting up factories inside the premises of the principal manufacturer.

It is stated that at the core of Industry 4.0 implementation, every asset (e.g. a machine, software, sensor or actuator, etc.), should be modeled via an Asset Administrative Shell (AAS). AAS is a knowledge structure that provides a description of the asset, its technical functionality and its relationships to other assets. It can be viewed as the bridge between the physical world and the IoT world or, in other words, the data model from where the digital twin stems from.

In this paper we present a methodology to create Asset Administrative Shells in a Mini Factories context. We argue that our approach, effectively embodies concepts related to Industry 4.0 in a modular way, allowing the much-desired agility.

By using our proposed methodology, Model Factory @ARTC has been able to create a seamless integration of assets into our factory of the future, demonstrating the connectivity and some of the enhanced capabilities of Industry 4.0.

© 2019 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of KES International.

Keywords: Automated Design and Configuration of Sensory Systems, Knowledge Acquisition, Knowledge Representation and Management, Model Factory, Industry 4.0

1. Introduction

The fourth industrial revolution is permeating heavily into the manufacturing landscape. National programs positioning countries in their respective regions, have been approved and put into action (\cite{1,14,15,19}). In Europe, the EU Commission launched the Factories of the Future program (2008 to 2020), committing an investment...
of €2.35B in a public–private partnership (PPP). The research domains include advanced manufacturing processes, digital, virtual and resource efficient factories. Germany, in particular, allocated a budget of €200M for Industrie 4.0, a strategic initiative launched in 2012, focusing on the digitalization of manufacturing [19]. In the case of the APAC region, one of the most relevant efforts is Singapore’s own Future of Manufacturing (FoM) [1], led by the Agency for Science, Technology and Research (A*STAR). In order to implement FoM, A*STAR has created two Model Factories, namely Model Factory @ SIMTech; and Model Factory @ ARTC (whose main research topics are depicted in Figure 1). These approaches intend to focus on the technologies, applications and the services that the Factory of the Future should possess.

Implementing the Model Factory, necessarily implies the realization that what may work in other regions may not be fully applicable to the local reality. At the same time considerations about standards, communication protocols and architecture perspectives must be considered. One of the most urgent matters, is that the factories of the future must satisfy the horizontal and vertical integration requirements through a seamless connectivity between assets [20].

In order to accomplish the aforesaid, Industry 4.0 recommends the use of the Asset Administration Shell (AAS), as the basic building block of the Factory of the Future ([2], [6], [27], [7]). AAS can be viewed as the bridge between a tangible asset and the IoT world or in other words the data model where the digital twin is based on.

Tangible assets or physical assets, are in fact machines, sensors, actuators and in general everything that can be part of a manufacturing plant and their processes. In such domain, the term “agile” has come to be one of the most requested features, and as a result, considerable research has been conducted to design manufacturing systems which embody manufacturing flexibility. It is in this context where the evolution of the supply-chain based concept of Mini Factories is relevant, as it allows a network-like interconnectivity between assets by setting up factories inside the premises of the principal manufacturer. ([11], [13], [5], [24]). As an example of the aforesaid fact, Muir et. al, [24], argue that “As the product transitions from one stage to the next, the factory design and programming should also transition from one embodiment to the next in order to achieve the best return on investment. Modularity of the factory components, highly flexible product transport mechanisms, and a high level of distributed intelligence are key characteristics of mini factory that enable this adaptation”.

In this paper we present a methodology to create Asset Administrative Shells in a Mini Factories context. We argue that our approach, effectively embodies concepts related to Industry 4.0 in a modular way. By using our proposed methodology, Model Factory @ARTC has been able to create a seamless integration of resources into our factory of the future, demonstrating their capabilities and connectivity.

The paper is organized as follows: in section 2, we present a brief overview on the state of the art, including relevant work and ground concepts where we based our approach. Section 3 presents our proposal of Asset Administrative Shells in Mini Factories context. Section 4 relates a case study from the Model Factory at ARTC, where our approach is being executed. Lastly in Section 5 we introduce some conclusions and present our future work.
2. Brief State of the Art

Researchers ([3], [5], [24]) have claimed for almost two decades that future factories, instead of being the gigantic environments we know today, will be composed by smaller more controllable mini-factories. Such approach is meant for quality control, flexibility and agility as well as the real-time needs of customers. A Mini factory, embodies two terms that are diametrical between them: (i) mass production, and (ii) customization. It can be argued then that though mini factories indeed “mass customization” can be achieved [23]. In the aforesaid line of thinking, mass customization allows for large numbers of customers to be reached, while still treating them as individuals that require products tailored to their specific needs. In this article we aim to show how the concept of mini factories is coming on age by looking at an example implementation through the eyes of Industry 4.0.

It is stated that at the core of Industry 4.0 implementation, every asset (e.g. a machine, software, sensor or actuator, etc.), should be modeled via an Asset Administrative Shell (AAS) [29]. An AAS is a knowledge structure that provides a description of the technical functionality of the physical object and its interactions. The aforesaid can be rationalized as the data representation of the physical world object as a digital twin.

AAS as presented by [17], are envisioned containing a number of views that represent different aspects, where each specific view is exposed depending on who is interacting with it (see Figure 2). According to [29], “The asset administration shell is thus made up of a series of submodels. These represent different aspects of the asset concerned; for example, they may contain a description relating to safety or security, but could also outline various process capabilities such as drilling or installation”.

![Figure 2. Structure of the AAS I4.0 component as described in [17]](image)

The AAS concept, stems from the idea that assets need to communicate amongst themselves and cooperate to execute tasks ([16], [4]). Key points of this concept are: (i) building upon the definitions of GMA FA 7.2. (ii) the suitability of the I4.0 Component for a wide range of life cycles in relation to the various partners of a value-added network. (iii) the possibility of locating the I4.0 Component in RAMI 4.0 (e.g. on the development side, on the production/ usage side, at a wide range of hierarchy levels). (iv) The possibility of operating I4.0-compliant communication equally for both active and passive connected assets. To do this, assets must go through different phases in which they check if they can speak to one another (language), identify one another (orientation), understand each other’s capabilities (become acquainted), agree (cooperate) and execute a task [16].
In their seminal work from 1997, Muir et. al. [13] set the foundations of Mini Factories and defined them as “a collection of mechanically, computationally, and algorithmically distributed robotic modules referred to as agents”. Such agents, are responsible for providing a minimum level of cooperation and communication in order to participate in the most basic operations of the mini factory. We believe that Muir’s definition falls in essence into the function that AAS seeks to deliver. By joining these two concepts, it is possible to show indeed an evolution on the implementation of the mini factories concept from the perspective of Industry 4.0.

There are a number of efforts made towards standardization, such as the Plattform Industrie 4.0 of the German government [18], and the Industrial Internet Consortium (IIC) [18]. The upcoming standards will enable the design of AAS-templates, which can later be customized for all kinds of assets by adding the required applications and data modules. As long as the asset has an accessible digital representation, even on a separate computer, the asset can be classified as an Industry 4.0 component and become a fully functional part of the mini factory.

Although there is some reported work on the implementation of AAS ([26], [25]), a replicable and easy to follow methodology using off the shelf solutions is still an open task.

Regarding to AAS implementations in the literature, Wenger. et. al [28], presented an approach to connect PLCs (Programable Logic Controllers) with Asset Administration Shells aimed towards automatic device configuration. The proposed mechanism implemented a non-blocking client for PLCs that queries information from a database server during run-time. The aforementioned database contains the AAS. Although it is stated that the proposed mechanism has been implemented for an IEC 61499 based RTE and that it can also be applied to IEC 61131 based RTES or any C++ application, it can be argued that the approach itself is case specific and it doesn’t address a full IoT platform approach.

One of the most interesting aspects of an AAS implementation is related to the highly semantic load that can be embodied into an AAS inspired data model. Grangel et. al, [8] introduced an approach to model AAS with the Resource Description Framework (RDF). The approach addresses the challenges of interoperable communication and machine comprehension in Industry 4.0 settings using semantic technologies. In the aforesaid work, it is shown how related standards and vocabularies, such as IEC 62264, eCl@ss, and the Ontology of Units of Measure (OM), can be utilized along with the RDF-based representation of the RAMI 4.0 concepts. The aforesaid work presented a very interesting approach on the semantics perspective, but it falls short on describing how to really make the connection between a physical asset and its virtual counterpart (the AAS). Last but not least, the Kaiserslautern (KL) Smart Factory project [10], has made significant contributions to the real-world implementation of AAS by even sharing their findings on a GIT repository. The KL approach presents examples of a Manufacturing Execution System (MES), and a motor between others. In this work, they start from the AAS modeling in Java Script Object Notation (JSON) and evolve programatically an AAS towards an OPC UA tree which allows the connection of the physical device to the AAS counterpart. This method although very relevant, does not deeply explain how to reach the scalability and agility that may be a requirement in a real-world scenario. In our work we get inspiration from their work and try to methodologically extend it by a rationalization of the process of creation and the use of an off-the-shelf commercial IoT platform that will be explained in sections 3 and 4 of this paper. As mentioned before, we believe that it is important to consider that in order to scale; a methodology is needed to systematically attack the issue of integrating AAS into existing (mostly brownfield) assets in a manufacturing shop floor. The integration is highly desired through an off-the-shelf Internet of Things (IoT) platform. Such IoT approach will in fact connect the physical and the virtual representation of the asset (in other words the machine with the AAS). We believe that it will be through iterations and re-factoring of a methodology to create the AAS in a manufacturing environment, that the benefits and potential pitfalls of this approach will be revealed. The lessons learned from the application of our methodology will tease out current technology and standard gaps that will need to be addressed in order to realize the complete picture of Industry 4.0.

3. Proposal for the implementation of Asset Administrative Shells in a Mini Factories context.

In this section we will introduce our approach for implementing AAS in mini factories. We will show how this methodology can be used and extended at the level of the AAS and how it reflects to the mini factory concept. It is commonly recognized that the concept of Cyber Physical Twin (CPT) is strongly linked to the fourth industrial revolution [20]. We also show that by following this simple approach, a CPT will be constructed and such elaboration
will bear a one to one functional relation to the physical asset it represents. The methodology can be divided into 3 different stages, described below.

**Stage 1: AAS Variables Definition:** This stage involves a discussion with domain experts on the relevant variables pertaining to the specific physical asset and its data submodels (the usage of the asset). Once the minimal amount of information to be collected is defined, these variables will be modelled in stage 2. The modeling of the AAS as stated by RAMI 4.0, contains two sections namely: *Property Definition* and *Property Characterization*, that define and specify the variables and their semantics. In the RAMI 4.0 AAS examples document [29] a variables definition and characterization based on hypothetical Manufacturing Execution System (MES) AAS is presented (see Figure 3).

![Figure 3. Example of an AAS for a “MES connection” submodel extracted from [29]](image)

Although we believe that *Property Definition* and *Property Characterization* are sufficient to describe the knowledge model of the AAS, we consider that the variables usage information is missing in the current recommended tables. The usage will give information about the data-in-use characteristics (e.g. storage, sampling time, throughput, etc.), needed for creating a tangible a connection from the physical world to the digital domain. Thus, in our approach we introduce the *Property Usage* that will add to the current model with three more columns:

- **KBITS_PER_SAMPLE:** indicating the size of the variable in Kilobits.
- **SAMPLING_TIME_MIN:** corresponding to the sampling time per minute.
- **THROUGHPUT_KB-MIN:** representing the rate of successful message delivery over the communication channel in Kilobits/minute.

With the addition of the Property Usage, one can calculate the required persistent storage in a local or cloud server, the network bandwidth usage and peaks and also design better strategies for data capture. One interesting outcome is that the cybersecurity aspects of the manufacturing infrastructure may be better designed and controlled, by having visibility to these quantities. The serialization of the Definition, Characterization and Usage can be done via Comma Separated Values (CSV) files.

**Stage 2: AAS modeling:** In this stage the methodology streamlines the information selected in the first stage CSV. The objective is to take the contained information and convert it into the AAS standardized model adding the header and body components [17]. The goal is to decouple the data model from the usability model. We’ve found that many users may be not familiar with the intricate semantics that an AAS structure possesses, but at the same time, they are knowledgeable about the variables needed and characteristics of such variables for producing a desired outcome.
At the same time, most of the generation of the AAS structure can be automatically performed using templates. In our approach we make use of the aforesaid AAS templates we have developed and focus on the submodel creation that comes from the definitions in stage 1.

The automatic creation based on templates produces a key-value JSON file (RDF or XML file is also possible), that adds mapping of the created submodel to AAS body. The process is completed via our own Python implementation which parses the CSV in stage 1 and creates the AAS serialized in JSON by adding the relevant corpus to the submodel. The serialization of the AAS is also possible in XML or RDF in case that ontology processing is desired in a later stage. The aforesaid approach will not only allow for a semantic representation of the data, but will provide a standardized approach to interact with AAS. For traceability, version control software can be used to track each version of the AAS.

Stage 3: AAS Connectivity with the real asset: This stage comprises the creation of the databases, communication protocol implementation and IoT integration that will be required to connect the physical to the digital world.

The process of database tables creation can be implemented in an automatic way as the variables are already defined in stage 1. The choice of the database is a particularly interesting topic. Although it is outside the scope of this paper, we recommend a time series database like Influx DB [9] due to the nature of most manufacturing processes that are mostly time based. Depending on the requirements, also an OPC UA tree for both server and client can be automatically created from the AAS model. In our example scenario we use an IoT platform RESTful API connectivity to do both the persistent storage (obtained from the database) and the communication protocol used to connect the real world to the digital twin that the AAS represents. Most of OPC UA SDKs have an automatic feature that allows the creation and consumption of variables expressed in XML-type files. We argue that for process data, the linking between the twin and the physical asset may lie in creating a service that populates the database tables, where this service feeds from an IoT connector. System architects considering DDS, can automate the DDS QoS mapping starting from the same key-value file. The benefits of using and IoT platform are evident as they offer ways to synchronize a physical asset via a smart gateway or Cyber Physical System using different low-level protocols that will map from the device specific data representation to more elaborated protocols. The case of Microsoft Azure IoT Hub [12] or PTC ThingWorx [21] are just few examples of the aforesaid approach which we recommend based on our experience. The following figure resumes the three stages described above (Figure 4).

4. Case study from the Model Factory at ARTC - Singapore

The Model Factory at ARTC provides a sandbox environment to deploy the methodology proposed in the previous section. The selected use case to apply our AAS approach is a gearbox factory which has three distinctive processes: Fabrication, Assembly and Warehousing (see Figure 5). The Fabrication/Turning process uses CNC machines, which perform machining on pre-treated shafts that will go into the gearbox. The assembly process is comprised of a manual work station and a robot work station, in which the gearbox is assembled. The warehousing process consists of racks
for finished product (Storage Racks), racks for consumables (Consumable Racks) and a self-guided autonomous transport unit (Transporter) to move the materials and products in the factory.

In this section we will go through the methodology to develop and deploy the AAS for a subset of machines/stations in the shop floor of the Model Factory at ARTC.

**Stage 1: AAS Variables Definition:** Working together with gearbox manufacturing experts, we were able to gather the key properties the gearbox factory needs to expose. In order to provide an example of the use of this approach, in this section, we will focus our attention on the robot station which is part of an assembly data model.

With the help of domain experts, we can break down each process of the production into a group of interlinked assets and the information they exchange to perform a task. This information is captured, in the assembly submodel (which applies to both robotic and manual stations) or in the robot submodel (only pertinent to robot stations). The information of each submodel is encoded into rows of properties in a CSV file. Selected properties from Robot Station and Assembly submodels (CSV files) are presented in Table 1.

Figure 5. Process representation of the data model for the gearbox factory. This section will focus on the robot station of the assembly sub-model (colored area).

<table>
<thead>
<tr>
<th>DATA MODEL</th>
<th>PROPERTY</th>
<th>PROPERTY DEFINITION</th>
<th>UNIT OF MEASURE</th>
<th>DATA TYPE</th>
<th>VALUE LIST</th>
<th>EXPRESSION LOGIC</th>
<th>EXPRESSION SEMANTIC</th>
<th>VIEW</th>
<th>CONTENTS</th>
<th>UNIT</th>
<th>SAMPLE</th>
<th>THROUGPUT</th>
<th>RPM</th>
<th>MIN</th>
<th>THROUGPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly</td>
<td>id</td>
<td>ID of the asset</td>
<td>NA</td>
<td>Integer</td>
<td>Any positive or Zero</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>STATIC</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assembly</td>
<td>ProductCount</td>
<td>Product count</td>
<td>Products</td>
<td>Integer</td>
<td>Any positive or Zero</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>DYNAMIC</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assembly</td>
<td>MaintenanceOrderId</td>
<td>Operator ID</td>
<td>NA</td>
<td>Integer</td>
<td>Any positive or Zero</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>DYNAMIC</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assembly</td>
<td>AGVPresence</td>
<td>AGV present in station</td>
<td>True, False</td>
<td>Boolean</td>
<td>True</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>DYNAMIC</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assembly</td>
<td>JobID</td>
<td>Job ID</td>
<td>NA</td>
<td>Integer</td>
<td>Any positive or Zero</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>DYNAMIC</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assembly</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robot Station</td>
<td>NextScheduledMaintenance</td>
<td>Scheduled date for next maintenance</td>
<td>NA</td>
<td>DateTime</td>
<td>...</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>STATIC</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robot Station</td>
<td>SerialNumber</td>
<td>Serial Number for the robot</td>
<td>NA</td>
<td>String</td>
<td>...</td>
<td>EQUAL</td>
<td>FUNCTIONAL</td>
<td>ALL</td>
<td>STATIC</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Properties for the Assembly and Robot Station data model (excerpt). Each row contains a property, with a property definition, property characterization and property usage.

**Stage 2: AAS modeling:** Having defined the submodels as CSV files, we can now parse them using our own Python implementation to generate the corpus of the AAS submodel, based on a template. The result is a JSON (or XML) file containing a header and body, where the body contains the submodel encoded in the CSV file. At this point, the JSON represents the AAS. The AAS creation step can be traced using version control software, so that whenever the domain requirements change and the CSV is modified, it’s possible to roll out a new version of the JSON file.

The AAS represented by the JSON file is then used in the PTC ThingWorx IoT platform. Using a REST API to automate the process, ThingWorx allows for the creation of types based on JSON files, where such types can later be used to create instances in the platform. For our implementation, and in the ThingWorx nomenclature, we use Data
Shapes (as types) and Data Tables (as instances). We create a child Data Shape (Robot Station type), extending the properties existing in a parent Data Shape (Assembly type, see Figure 6), thus replicating the submodel structure of our data model in the ThingWorx platform.

Starting from the CSV files created in Stage 1, we have created and instantiated the Robot Station type in the ThingWorx IoT platform. At this point we are ready to connect the Robot Station instance with information sources to populate it programmatically.

Stage 3: AAS Connectivity with the real asset: At this point of our case scenario, the Robot Station instance only contains static data and is yet to be populated with dynamic data. ThingWorx allows for the creation of services that can be programmatically executed on a timer to retrieve data from dynamic data sources. KEPServerEX (another software from PTC) [22] acts as a connector to the real world, allowing the usage of different communication protocols to interact with physical assets, as well as connecting to ThingWorx. Additionally, ThingWorx allows for the creation of persistent storage in databases using the defined submodel. By leveraging on ThingWorx IoT platform and KEPServerEX, through the usage of the standardized JSON AAS, we achieve a digital twin of the physical asset (see Figure 7).

Through the ThingWorx platform, having completed the three stages of the methodology, the AAS gives us an instantaneous virtual view of the robot station status, providing information that other systems and workers need to collaborate with it. By having a common system of representation, it’s straightforward to imagine and develop systems that can leverage on the different AAS to empower factory management and operations (see Figure 8).
From a technical literacy point of view, the approach requires understanding of programming languages for the developer of the serialization and deployment software. However, for a factory user to consume the applications, the requirements are basic knowledge of spreadsheets and basic understanding of the IoT platform in-place (ThingWorx in our example) to verify the deployment.

5. Conclusions and future work

In this paper we have introduced our methodology to create Asset Administrative Shells in a Mini Factories context. We argue that our methodology can be applicable into a manufacturing facility who intends to adopt a pathway towards Industrie 4.0. In order to show our methodology in action, we have presented a case study which was implemented at the Model Factory @ARTC. The aforesaid case study relates a full life cycle implementation from the physical asset to its digital counterpart represented by the AAS. Efforts were placed in crafting a methodology as automated as possible. The reason behind this is not speed, although its critical to scale the deployment, but to smooth out an integration and on-boarding process that would otherwise be on a ‘case by case’ basis, leading to mismatched information and lack of predictability of available metrics. Together with the property usage information, this approach not only allows the factory manager to know what information will be available for each asset, but also exactly how often the information will be updated. As future work, we are working on the full automation of the different stages of our suggested approach. The reason why is because the selected tools for our implementation offer application interfaces that will allow by coding, the automatic generation and updating of an AAS leading to better and easier maintenance of it. Also, we are currently working into using the presented approach in new domains such as: FMCG (Fast Moving Consumer Goods).

Acknowledgment

This research is supported by the Agency for Science, Technology and Research (A*STAR) under its Advanced Manufacturing & Engineering (AME) Industry Alignment Funding – Pre-positioning funding scheme (Project No: A1723a0035)

References


