










Model-Driven Engineering in Digital Thread Platforms: A Practical Use Case and Future Challenges

Hafiz Ahmad Awais Chaudhary^{1,4} , Ivan Guevara^{1,4} , Jobish John^{2,4} ,
Amandeep Singh^{1,5} , Amrita Ghosal^{1,4} , Dirk Pesch^{2,4} ,
and Tiziana Margaria^{1,3,4,5} 

¹ University of Limerick, Limerick, Ireland

{ahmad.chaudhary,ivan.guevara,amandeep.singh,amrita.ghosal,
tiziana.margaria}@ul.ie

² University College Cork, Cork, Ireland

{j.john,d.pesch}@cs.ucc.ie

³ Lero - The SFI Software Research Centre, Limerick, Ireland

⁴ Confirm - Centre for Smart Manufacturing, Castletroy, Ireland

⁵ Centre for Research Training in Artificial Intelligence (CRT AI), Cork, Ireland

Abstract. The increasing complexity delivered by the heterogeneity of the cyber-physical systems is being addressed and decoded by edge technologies, IoT development, robotics, digital twin engineering, and AI. Nevertheless, tackling the orchestration of these complex ecosystems has become a challenging problem. Specially the inherent entanglement of the different emerging technologies makes it hard to maintain and scale such ecosystems. In this context, the usage of model-driven engineering as a more abstract form of glue-code, replacing the boilerplate fashion, has improved the software development lifecycle, democratising the access to and use of the aforementioned technologies. In this paper, we present a practical use case in the context of Smart Manufacturing, where we use several platforms as providers of a high-level abstraction layer, as well as security measures, allowing a more efficient system construction and interoperability.

Keywords: Edge computing · Smart Manufacturing · Digital Thread · Model driven engineering

1 Introduction

In the actual data-centric era, where the shift towards distributed and ubiquitous architectures demands increasing computing capabilities, cloud-based processing represents a bottleneck. This is due to the increasing amount of devices taking part in the systems, which are often systems of systems. According to the Cisco Annual Report 2018–2023 [16], IoT devices will account for 50% (14.7 billion)

of all global networked devices by 2023, having an impact in the workload processing and forcing companies and organizations to look for more cost-effective and efficient alternatives. In this context, since the data is generated at the edge of the network, i.e., by the IoT devices, it would be more efficient if the processing of the data also happens at the edge. This is called Edge computing [29]. Edge computing plays a significant role in the Industrial IoT (IIoT) sector, lowering the cost of data transport, decreasing latency and improving the overall efficiency of the architecture [32]. This does not mean that cloud and edge computing are incompatible, rather they complement each other, allowing us to have more tools to confront these issues and be able to improve the overall performance in a balanced and customized way.

Based on physical configurations and functional requirements in the context of smart manufacturing, there are many standard architectures for the implementation of IoT based systems. ISA-95[IEC 62264-1:2013] [21] is one of the international standard for enterprise control system integration that defines both physical arrangements and functional hierarchies from device to device communication to functional management at different granular levels. FIWARE [27] platform is an EU initiative towards the development of smart applications in manufacturing with a set of standardized APIs. Similarly EdgeX Foundry [17] is an open source standardized interoperability framework for IIoT Edge computing and Gaia-X [13] is a European standard for the development of next generation of data oriented infrastructures.

Shifting massive amounts of data towards the edge has also a downside: it requires to reconfigure the computation architecture in order to leverage computations on the edge components, and, as a consequence, also the ability to orchestrate the different heterogeneous responses from each SDK brought in by devices of the ecosystem. This is a massive ask to the ability to integrate not just data, but processes whose bits and pieces are often buried in the SDKs. Another challenging task is the integration of the edge intelligence [20], where the cost of delivering such solutions could be high if the advantages and limitations of this technology are not taken into account.

Model driven development (MDD) is one of the key approaches to develop heterogeneous systems from the conceptual modelling design to automated model-to-code transformations [25]. The main goal [28] of MDD is to develop rapid applications, that are flexible and adaptive to continuously changing requirements. The goal of this case study is to rely on model-driven capabilities as far as it is possible and convenient, i.e., without forcing the entire ecosystem to be integrated in a single platform. We use here the EdgeX Foundry platform as a middleware for IIoT components for data acquisition from heterogeneous sensors. Additionally, we use three low-code platforms for the development of functional pipelines: Tines for notifications among systems, Pyrus for data analytics and the DIME platform for process modeling and data reporting, empowering prototype-driven application development. All three follow to different extents the XMDD paradigm [24], where the technical details of the communication with a component or subsystem are encapsulated in high-level models, and this abstraction

is useful in order to rapidly bootstrap a workflow and enable non-expert programmers to responsibly and directly participate in the software development cycle.

In the following, Sect. 2, discusses the industrial use-case with its system architecture, Sect. 3 covers the secure access policies and encryption techniques, followed by our conclusions and reflections in Sect. 4.

2 Industrial Use-Case: Safe Operation of Machines

In this section, we detail an industrial use case associated with the safe operation of a machine. Most manufacturing industries are equipped with complex machines on their shop floor, such as computer numerical control (CNC), coordinate measuring machines (CMM), 3D printers, etc. They are monitored and controlled through an industrial automation network in addition to the human-machine interface. A safe, healthy operating environment is essential in such factory and laboratory areas. The machine operators are expected to comply with several health and safety measures, such as wearing gloves, glasses, boots, aprons and hats where opportune. For example, in certain areas on the production floor protection measures such as boots and glasses are mandatory.

The machine operators however may not always comply and follow all the safety measures, incurring higher safety risks. This is a recognized occupational hazard, and companies and organizations have high interest in minimizing such risks and hazards. In this direction, an automatic system [11] consists of a wearable devices and an NVIDIA Jetson board is proposed in an industrial scenario for monitoring activities of personals and the behaviors of robotic systems.

One way to address non-compliance is by monitoring the production floor with cameras that continuously monitor these areas for proper behaviour along health and safety guidance, and also for assistance in case of need. Several of these machines have inbuilt sensors that measure various parameters such as vibrations, temperature, pressure etc. In many cases, these parameters can be used as a marker to identify the health of machines and tools. Several industries follow additional digitization strategies by employing additional IIoT sensing modules. The machine/tool health status inferred from these sensor data, combined with the camera-based monitoring, can then ensure that proper health and safety measures for both machines and operators are followed on an industrial workyard. Security measures have to be taken as well, here we concentrate on attribute-based cryptography as a means to ensure that there are no leaks of business or privacy sensitive data.

2.1 Architecture of the Use Case

Figure 1 shows the system architecture of the case study. Several different sensors installed in critical locations across the factory workyard monitor the working conditions on the floor of the Industrial Setup, on the left. Cameras are also installed to visually supervise the safety of the workers. When the workers

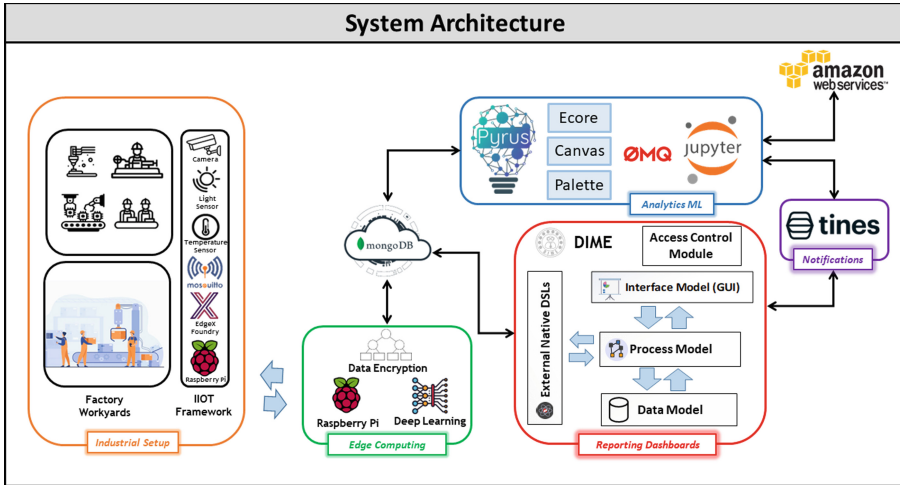


Fig. 1. System architecture of the case study

are working with critical equipment, the machine and environmental parameters/conditions, e.g. vibrations, temperature, light, etc. are recorded using the deployed sensors, belonging to the IIoT Framework. In addition, the camera feed is also used to record the workyard at random time intervals (Factory Workyard). These sensors and cameras are connected by an Edge-controlling device, here a Raspberry Pi, which securely sends the collected data to another Raspberry Pi that acts as an Edge-computing and collection node for all the devices across the factory workyard. This computing/collection node is responsible for communication with all the data nodes and for sending this data to a reliable database, in our case a MongoDB Atlas instance. It also performs deep learning-based computations at the edge (the Edge Computing system). Since the computing/collection node is a Raspberry Pi, its computing capacity is limited and is only used for computations that require an extra layer of privacy, accountability and scrutability. We consider here the facial recognition of the workers for an attendance call, or the identification of workers present in the workyard.

From MongoDB, the data is accessed by the Analytics/ML system. There, Pyrus is responsible for periodically running ML pipelines at fixed intervals by communicating with the Amazon Rekognition API to detect the PPE/safety equipment as shown in Fig. 2. The results from these pipelines are a number of reports on whether all workers are satisfying the safety requirements of their respective tasks. They are sent to the MongoDB Atlas database for remote secure access by workyard supervisors. When the results are stored on MongoDB, a notification about this event is sent via the Tines automation pipeline as shown in Fig. 3. Triggered by the Tines web-hook notification, a web application implemented in DIME fetches the data from MongoDB and generates reporting dashboards that can be viewed by the supervisors to make critical

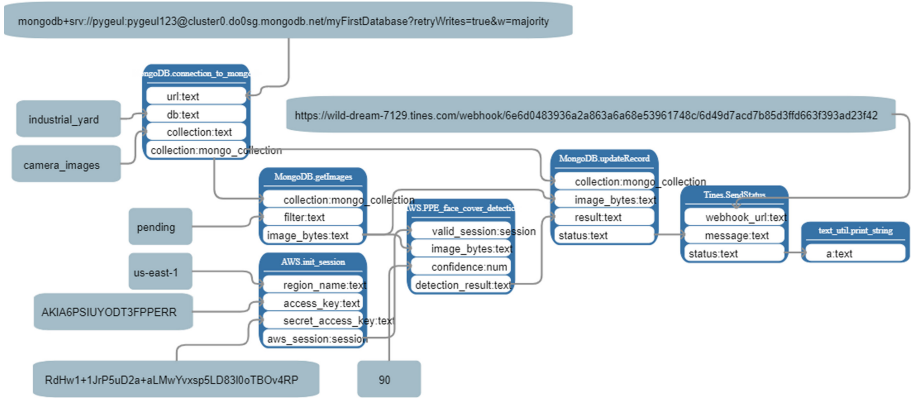


Fig. 2. Pyrus PPE detection pipeline

decisions, creating adequate Reporting Dashboards. The Access Control Module implements an attribute-based mechanism that abstracts from individuals and specific entities, basing the access through specific cryptography on attributes, that can be roles or other elements of a profile.

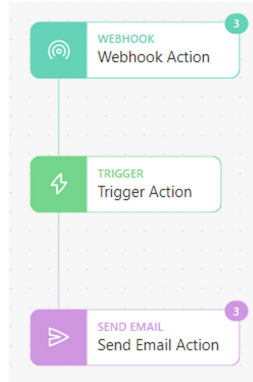


Fig. 3. Tines automation pipeline

2.2 The IT Ecosystem: Tools and Technologies

In this section, we briefly discuss the individual tools and technologies that form the heterogeneous ecosystem involved in this case study.

Raspberry Pi. The devices chosen for collection, computation and sharing of data in this research were Raspberry Pis due to their cost effectiveness, support for all devices used, support for Python and Linux, and availability of many

interface options onboard. Two types of Raspberry Pis cover the two roles in this setup, as shown in Sect. 2.1:

- The **Edge-controlling device** is a Raspberry Pi Compute Module 3 (CM3) board connected with a StereoPi V0.9 carrier board [5] that is capable of connecting to two cameras on the same board using ribbon cables. The StereoPi device is running the *StereoPi Livestream Playground* v2 (SLP) image [6] that provides a consumer-friendly administration panel (similar to those in WiFi routers) without the need of setting up separate peripherals such as keyboard, mouse, monitor, etc., simplifying the setup and deployment. The SLP image also allows streaming the camera feed [6] to a UDP client through private IP address and port capturing using tools like *Gstreamer* [2]. Using this feature, the video feed is sent to the Edge-computing and collection node, from where it can be processed/stored according to the needs.
- The **Edge-computing and collection device** is a Raspberry Pi 4, model B with 4 GB RAM. It is the latest series in the Raspberry Pi series of single board computers with a high-performance 64-bit quad-core processor.

Sensors and Devices. The most commonly used sensors for the machine or tool maintenance are vibration and temperature sensors. *EPH-V11*, *EPH-V17*, *EPH-V18*, *EPH-T20* [18] are some of the widely used sensors that provide vibration and temperature data over wireless communication. They report data over ModBus, which is one of the most widely used standard for industrial communication.

The cameras are Raspberry Pi *High Quality Camera* (HQCam) Modules with CGL interchangeable lenses [4]. The HQCam modules have 12.3 megapixel cameras, 7.9 mm diagonal image size, support 12-bit RAW footage, have adjustable back focus and are compatible with C/CS mount lenses. The CGL lenses are 3 megapixel 6 mm HD CCTV lenses with inbuilt IR filter. The camera modules are connected to the Raspberry Pi through 200 mm ribbon cables.

Edgex Foundry. EdgeX Foundry [17] is an open source, vendor neutral, flexible, inter-operable, software platform at the edge of the network, that interacts with the physical world of devices, sensors, actuators, and other IoT objects. It is used as a middleware integration and virtualization platform, as it directly connects with the IoT devices and exposes high-level services through a REST API. We have described the EdgeX integration in DIME in [14, 19].

DIME. The DIME [12] integrated modelling environment is a development environment to easily design, develop and deploy Web Applications in a low-code/no-code manner. It supports different model types (GUI, process and data models) that address different aspects of a Web Application. Built-in checks are supported both at the model and the project level for the purpose of debugging, as well as one-click code generation and deployment. Its *External Native DSL* layer, described in detail in [15], provides the flexibility to extend the platform

capabilities with external services and platforms. This is the capability that we exploit for the integration of external devices, tools and platforms. We have not integrated everything in DIME for a number of reasons. First of all, DIME database and front-end components do not support the complex data types like video streams. In addition, we wanted to show the interoperability among the different heterogeneous tools and technologies.

Pyrus. The Pyrus [33] web-based no-code collaborative platform for data analytics provides the support of basic data manipulation and analytics operations in a model-driven fashion. It represents the individual capabilities as a collection of taxonomically grouped SIBs (Service-Independent Building blocks) in its Ecore (name of SIBs palette) section. On the backend, Pyrus communicates over the ZeroMQ protocol [8] with the connected Jupyter Hub, initially for functions discovery of the available SIBs, then at runtime to call and execute the advanced Python programs that the SIBs represent as proxies. Technically, the Pyrus workflows are data-flow orchestrations of SIBs.

Tines. The Tines [7] story board is a no-code automation tool that was initially built for automating workflows (called stories) in the domain of security. It also works in a low-code approach, with seven ‘actions’, i.e. generic components that are similar to highly parametric SIB templates in the world of DIME and Pyrus. Its webhook actions are here used as triggers, and its notification capabilities are domain-independent, so we use here a small story that implements an automation workflow for notifications.

Amazon Rekognition. The Amazon Rekognition [1] pre-trained deep learning API provides the capabilities of images and video analytics. Its models are trained by Amazon on billions of public photos from Amazon Prime and optimised for specific use-cases such as face detection, PPE detection, etc. We use this API in our Pyrus pipelines for the detection of safety-equipment among staff working in the factory workyard. The advantage of using Amazon Rekognition API in our Pyrus pipelines is that it does not require any additional setup and it worked out-of-the-box.

MongoDB. The MongoDB Atlas [3] is a cloud-based NoSQL database service that is used for high-volume data storage of semi-structured or unstructured data. In contrast to the structured records and tables used by relational databases, Atlas stores the data in the form of documents and collections which support the flexibility of different non-structured data types. The Atlas database is also scalable for Big Data storage with support for clusters that can store millions of documents. We use it here to store all the observational and processed data from different sensors, edge devices and compute systems. Atlas is pre-integrated in Tines, which provides no-code SIBs called ‘actions’ for easy communication with Atlas instances.

3 Access Control Using Attribute Based Encryption

To facilitate a fine-grained access control in the smart factory to which the workyard belongs, we utilize a public-key encryption, Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [10] for this particular use-case. The CP-ABE algorithm allows for identification of the ciphertexts with access structures and the private keys with attributes. Whenever a message is encrypted using CP-ABE, it generates a ciphertext based on the condition that only a single user who is the owner of the specific attributes and satisfies the access structure will be able to produce the private key, and thereby decrypt the message. One of the highlights of CP-ABE is that it permits the definition of top-level policies, and is particularly suitable in scenarios where an individual wants to restrict the access to a specific information only to a subset of users within the same broadcast domain [9]. Another aspect of CP-ABE is that it is robust by design against collusion attacks [30]. A CP-ABE scheme consists of the following four basic algorithms:

- $\text{SETUP}()$. This algorithm generates the public key pk_c and master key mk_c .
- $\text{KEYGEN}(mk_c, \text{Attr}_c)$. This algorithm takes mk_c and the user attribute list Attr_c as input and returns a private key pv_c of user C .
- $\text{ABE}_{pk_c, w}(m)$. The encryption algorithm takes pk_c , an access policy w over the pool of attributes, and sensor reading m as input. It returns a ciphertext that can only be decrypted by a user that possesses a set of attributes Attr_c such that Attr_c satisfies w .
- $\text{ABD}_{pv_c}(\mathcal{C})$. The decryption algorithm takes pk_c , pv_c of user C and the ciphertext \mathcal{C} as input. It outputs the plaintext m if and only if the user Attr_c satisfies w .

For our industrial use case, we define a secure access policy along the lines of CP-ABE as follows: we consider three attributes in our use case scenario: *Sensor Examiner* (SE), *Video Analyst* (VA) and *Decision Manager* (DM) for three departments in our industrial use-case setting. The main characteristics of the three departments are described as follows:

- We first consider the Technical Fault Monitoring Department (TFMD). The primary task of TFMD is to monitor the readings of the three sensors (temperature, pressure, vibration) and it has access to the data collected by these three sensors. We assign SE attribute to TFMD.
- We then consider the Safety Surveillance Department (SSD). The main responsibility of SSD is to analyse the videos captured by the video camera. Thereby, the SSD has access to the readings of the video camera. We assign VA attribute to SSD.
- Finally, we consider the Operation Management Department (OMD). OMD takes the final call for the need of generating an alarm if any emergency occurs. Emergencies are reflected through the readings of the corresponding sensors and/or the videos. We assign DM attribute to OMD.

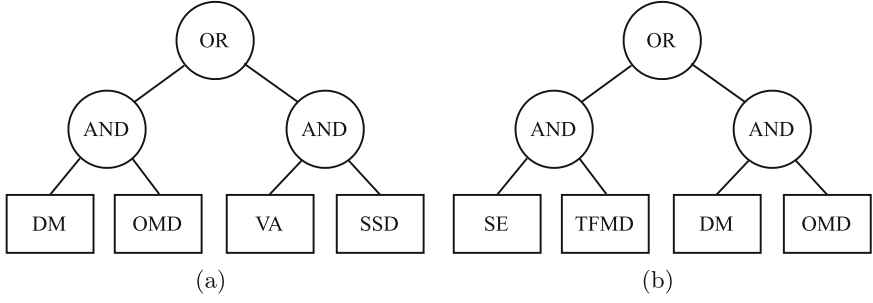


Fig. 4. Structure of access policies for our scheme: (a) Video surveillance, (b) Sensor reading.

3.1 Bilinear Map

Our CP-ABE is based on a bilinear map. Let \mathbb{G} and \mathbb{G}_T be two multiplicative cyclic groups of prime order p . Let g be a generator of \mathbb{G} and e be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. The bilinear map e has the following properties:

- Bilinearity: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degenerate: $e(g, g) \neq 1$.

Here, \mathbb{G} is a bilinear group if the group operation in \mathbb{G} and the bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ are both efficiently computable. It is worth noting that the map e is symmetric as $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

3.2 Decision Tree

Let \mathcal{T} be a decision tree representing an access structure. Figure 4 illustrates the simple decision trees that are generated from the access policies defined using the specific attributes and their entities. The access policies define which entities have access to specific data generated by the devices. As mentioned previously, we intend to allow fine-grained secure access control based on attributes, and for this we leverage public-key encryption, i.e., CP-ABE. In our case, the policies define that in a normal context, the Video Surveillance access policy is that either a DM belonging to the OMD department or a VA belonging to the SSD department can have access. Similarly, the Sensor reading policy is that either a SE belonging to the TFMD department or a DM belonging to the OMD department can have access.

As we complete the use case with the access to other elements of the ecosystem, the set of policies will become more complex, as different entities will be added, with typically partially overlapping rights. Only the individuals with the specific attributes will be able to access the data and/or perform operations.

3.3 Our Construction

We now provide our main construction of the fine-grain access control approach.

SETUP. The setup algorithm chooses a bilinear group \mathbb{G} of prime order p with generator g . It then selects two random exponents $\alpha, \beta \in \mathbb{Z}_p$. The public key is determined as:

$$pk_c = \mathbb{G}, g, h = g^\beta, f = g^{1/\beta}, e(g, g)^\alpha$$

and the master key mk_c is set as (β, g^α) .

KEYGEN($mk_c, Attr_c$). Our key generation algorithm takes the master key mk_c and a set of attributes $S = \{SE, VA, DM\}$ associated with the department as inputs and provide a private key that identifies with that set. This algorithm initially chooses a random $r \in \mathbb{Z}_p$, and next random $r_j \in \mathbb{Z}_p$ for each attribute $j \in S$. In our algorithm, the private key is generated as follows:

$$pv_c = (D = g^{(\alpha+r)/\beta}, \forall j \in S : D_j = g^r \cdot H(j)^{r_j}, D'_j = g^{r_j}),$$

where H is a hash function, $H : \{0, 1\}^* \rightarrow \mathbb{G}$.

ABE $_{pk_c, w}(m)$. Our encryption algorithm encrypts a message m under the decision tree structure \mathcal{T} . Beginning with the root node N , our algorithm chooses a random $\delta \in \mathbb{Z}_p$. Let L be the set of leaf nodes in \mathcal{T} . The ciphertext is generated by giving the decision tree \mathcal{T} and determining:

$$\mathcal{C} = (\mathcal{T}, \bar{C} = me(g, g)^{\alpha s}, C = h^s, \forall y \in Y : C_y = g^{q_y(0)}, C_{yp} = H(att(y))^{q_y(0)}),$$

where the function $att(x)$ signifies the attribute associated with the leaf node x in the decision tree \mathcal{T} . In the above equation, our algorithm generates random value s to calculate the shared value $q_y(0)$ for each attribute in the decision tree \mathcal{T} using linear secret sharing. In CP-ABE, as private keys are generated randomly using the decision tree \mathcal{T} , it thus prevents collusion attack.

ABD $_{pv_c}(\mathcal{C})$. Our decryption algorithm executes recursively. Here, we present the simplest form of our decryption algorithm. If x is a leaf node, we let $i = att(x)$. If $i \in S$, our algorithm computes message m from the ciphertext \mathcal{C} using pv_c as follows:

$$m = \frac{e(D_i, C_x)}{e(D'_i, C'_x)}.$$

We refer the reader to [10] for further details about the above computations.

4 Conclusions and Reflections

In terms of the Digital Thread [23], work on the overall lifecycle ecosystem that supports the smooth interoperation of a physical facility (like a machine, a factory or even a supply chain) and its direct or derived digital components (data and capabilities, but also processes, decisions, security) are still topic of ongoing research and are under-development. The integration is often done ad hoc, and

successful platform attempts are domain and layer-specific, like EdgeX for IoT middleware. In this case study we have decided not to integrate everything in DIME, for a number of reasons.

First of all, we wanted to leverage the different integrations that already exist (like the IoT sensors in EdgeX, MongoDB in Tines, AWS in Pyrus) and preexisting communication routes (like EdgeX and MongoDB) and some of the workflows, and see how these islands of integration could be further brought together to a complex scenario. The experience is that there is still a considerable need of adaptation and testing, as for example the networks and protocols (which depend on configurations) and the specific software versions do matter.

We also decided against a full integration in DIME as for example the current DIME data types do not support video streams, therefore a direct integration of camera output would not be feasible at present. In this sense, we play to the individual strengths of the different platforms and integration and abstraction/virtualization approaches.

We wanted to show how a quite complex system of systems can come together in quasi-realistic settings. We also showed that it allows a piece-wise integration using four different platforms, three of whom using models and low code approaches (Tines, Pyrus and DIME). Of those, Pyrus and DIME are Cinco-products [26] and XMDD [24] approaches, whereby Pyrus is a web-based data-flow specialized tool for data analytics, while DIME is a much more complex and general-purpose tool with complex interdependent model types. Taken together, this is a step towards enabling a heterogeneous analysis and verification for distributed systems, as in [31], and the possibly hierarchical organization of reusable portions of logic in terms of features [22].

The inclusion of security is at the moment still simple: it is based on attributes that are roles. In this sense it is de facto similar to Role Based Access Control, but the use of attributes and their connection to the encryption make it more flexible (as one could also consider more attributes that are context-dependent), and more secure.

Acknowledgements. This project received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Smart 4.0 Co-Fund, grant agreement No. 847577; research grants from Science Foundation Ireland (SFI) under Grant Number 16/RC/3918 (CONFIRM Centre), 2094-1 (Lero, the Software Research Centre) and 18/CRT/6223 (CRT-AI). We are also thankful to Dr. Kevin Moerman (National University of Ireland, Galway) for providing us the equipment and support for the StereoPi and cameras setup.

References

1. Amazon Rekognition | automate your image and video analysis with machine learning. <https://aws.amazon.com/rekognition/>. Accessed May 2022
2. GStreamer | Open Source Multimedia Framework. <https://gstreamer.freedesktop.org/>. Accessed May 2022
3. MongoDB Atlas Database | Multi-Cloud Database Service. <https://www.mongodb.com/atlas/database>. Accessed May 2022

4. Raspberry Pi High Quality Camera. <https://www.raspberrypi.com/products/raspberry-pi-high-quality-camera/>. Accessed May 2022
5. StereoPi - DIY stereoscopic camera based on Raspberry Pi. <https://stereopi.com/>. Accessed May 2022
6. StereoPi Wiki Main Page. <https://wiki.stereopi.com/>. Accessed May 2022
7. Tines | no-code automation for security teams. <https://www.tines.com/lessons/storyboard/>. Accessed May 2022
8. Zeromq | an open-source universal messaging library. <https://zeromq.org/>. Accessed May 2022
9. Ambrosin, M., Busold, C., Conti, M., Sadeghi, A.-R., Schunter, M.: Updicator: updating billions of devices by an efficient, scalable and secure software update distribution over untrusted cache-enabled networks. In: Kutyłowski, M., Vaidya, J. (eds.) ESORICS 2014. LNCS, vol. 8712, pp. 76–93. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11203-9_5
10. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (SP 2007), pp. 321–334. IEEE (2007)
11. Boldo, M., et al.: Integrating wearable and camera based monitoring in the digital twin for safety assessment in the industry 4.0 era. In: Margaria, T., Steffen, B. (eds.) ISoLA 2022, LNCS 13704, pp. 184–194. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-19762-8_13
12. Boßelmann, S., et al.: DIME: a programming-less modeling environment for web applications. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016. LNCS, vol. 9953, pp. 809–832. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47169-3_60
13. Braud, A., Fromentoux, G., Radier, B., Le Grand, O.: The road to European digital sovereignty with GAIA-x and IDSA. IEEE Network **35**(2), 4–5 (2021). <https://doi.org/10.1109/MNET.2021.9387709>
14. Chaudhary, H.A.A., Guevara, I., John, J., Singh, A., Margaria, T., Pesch, D.: Low-code internet of things application development for edge analytics. In: Camarinha-Matos, L. M., et al. (eds.) Internet of Things. IoT through a Multi-disciplinary Perspective, IFIP IoT 2022, IFIP AICT 665, pp. 1–20 (2022). https://doi.org/10.1007/978-3-031-18872-5_17
15. Chaudhary, H.A.A., Margaria, T.: DSL-based interoperability and integration in the smart manufacturing digital thread. Electron. Commun. EASST **80** (2022)
16. Cisco: Cisco, March 2022. <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html>
17. EdgeX Foundry: The preferred edge IoT plug and play ecosystem - eabled open source software platform. <https://www.edgexfoundry.org/>. Accessed May 2022
18. Erbesd instruments: Condition monitoring & industrial automation. <https://www.erbesd-instruments.com/wireless-vibration-sensors/>. Accessed May 2022
19. Guevara, I., Chaudhary, H.A.A., Margaria, T.: A low-code proposal for a rule-based engine integration in a digital thread platform context. In: International Manufacturing Conference IMC, vol. 38 (2022)
20. Guevara, I., Chaudhary, H.A.A., Margaria, T.: Model-driven edge analytics: practical use cases in smart manufacturing. In: Margaria, T., Steffen, B. (eds.) ISoLA 2022, LNCS 13704, pp. 406–421. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-19762-8_29
21. Enterprise-control system integration. Standard, International Organization for Standardization, May 2013

22. Karusseit, M., Margaria, T.: Feature-based modelling of a complex, online-reconfigurable decision support service. *Electron. Notes Theor. Comput. Sci.* **157**(2), 101–118 (2006)
23. Margaria, T., Schieweck, A.: The digital thread in industry 4.0. In: Ahrendt, W., Tapia Tarifa, S.L. (eds.) IFM 2019. LNCS, vol. 11918, pp. 3–24. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34968-4_1
24. Margaria, T., Steffen, B.: Service-orientation: conquering complexity with XMDD. In: Hinchey, M., Coyle, L. (eds.) *Conquering Complexity*, pp. 217–236. Springer (2012). https://doi.org/10.1007/978-1-4471-2297-5_10
25. Mellor, S.J., Clark, T., Futagami, T.: Model-driven development: guest editors' introduction. *IEEE Softw.* **20**(5), 14–18 (2003). ISSN 0740–7459
26. Naujokat, S., Lybecait, M., Kopetzki, D., Steffen, B.: CINCO: a simplicity-driven approach to full generation of domain-specific graphical modeling tools. *Int. J. Softw. Tools Technol. Transfer* **20**, 1–28 (2018). <https://doi.org/10.1007/s10009-017-0453-6>
27. Salhofer, P., Joanneum, F.: Evaluating the FIWARE platform: a case-study on implementing smart application with FIWARE. In: *Proceedings of the 51st Hawaii International Conference on System Sciences*. vol. 9, pp. 5797–5805 (2018)
28. Sanchis, R., García-Perales, Ó., Fraile, F., Poler, R.: Low-code as enabler of digital transformation in manufacturing industry. *Appl. Sci.* **10**(1), 12 (2020)
29. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. *IEEE Internet Things J.* **3**(5), 637–646 (2016). <https://doi.org/10.1109/JIOT.2016.2579198>
30. Song, H., Yin, F., Han, X., Luo, T., Li, J.: MPDS-RCA: multi-level privacy-preserving data sharing for resisting collusion attacks based on an integration of CP-ABE and LDP. *Comput. Secur.* **112**, 102523 (2022)
31. Steffen, B., Margaria, T., Claßen, A., et al.: Heterogeneous analysis and verification for distributed systems. *Softw. Concepts Tools* **17**, 13–25 (1996)
32. Wang, X., Han, Y., Leung, V.C., Niyato, D., Yan, X., Chen, X.: Convergence of edge computing and deep learning: a comprehensive survey. *IEEE Commun. Surv. Tutorials* **22**(2), 869–904 (2020)
33. Zweihoff, P., Steffen, B.: Pyrus: an online modeling environment for no-code data-analytics service composition. In: Margaria, T., Steffen, B. (eds.) *ISoLA 2021*. LNCS, vol. 13036, pp. 18–40. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-89159-6_2

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

