5-11-2023

EMBRACING CONCEPTUAL MODELLING TO ENABLE THE MEDIATION ROLE OF ENTERPRISE INFORMATION SYSTEMS

Andrei Chis
Babes-Bolyai University, Faculty of Economics and Business Administration, andrei.chis@econ.ubbcluj.ro

Ana-Maria Ghiran
Babes-Bolyai University, Faculty of Economics and Business Administration, anamaria.ghiran@econ.ubbcluj.ro

EMBRACING CONCEPTUAL MODELLING TO ENABLE THE MEDIATION ROLE OF ENTERPRISE INFORMATION SYSTEMS

Research Paper

Andrei Chiș

Babeș-Bolyai University, Cluj-Napoca, Faculty of Economics and Business Administration, Romania, andrei.chis@econ.ubbcluj.ro

Ana-Maria Ghiran

Babeș-Bolyai University, Cluj-Napoca, Faculty of Economics and Business Administration, Romania, anamaria.ghiran@econ.ubbcluj.ro

Abstract

Nowadays, digital environments are critical regarding the way enterprise operations are conducted and how enterprises are being developed. Traditional information systems used by enterprises need to evolve to accommodate the shift of their role from representation to mediation, role that has been recently conceptualized but not sufficiently researched to identify means that enable its operationalization across domains and purposes. This work identifies what are the key enablers and how should these be applied regardless of the domain of interest. For this, our paper showcases as a proof-of-concept artifact that operationalizes the before mentioned new role of the IS. Our proposal relies on Conceptual Modelling together with all the other related enablers like Digital Twins, metamodelling platforms or model-driven operation. Our example concerns the modern organization that continuously aims to become more innovative and gain competitive advantage therefore, it employs smart devices like drones in different business operations.

Keywords: Conceptual Modelling, Digital Twins, Physical Twins, Smart drones.

1 Introduction

The work presented in this paper describes the development of a new enterprise management system, up to the proof-of-concept level, that aims to bridge the gap between digital and physical environments in Industry 4.0 related scenarios. Concretely, we developed a process modelling tool that allows roundtrip engineering capabilities between environments, allowing digital environments to capture the reality and also to shape the way in which operations take place in the physical world.

Our research goal is to develop a means that allows digital environments to act as controllers for physical environments in variate business scenarios, extending the applications of IoT-oriented technologies. The physical world is, in this way, also a result of the decisions made in the digital world, phenomenon that was called ontological reversal, as presented by Baskerville et al. (2020). The layer provided by our work allows IS to *represent* real world systems and to *influence* their operation, providing more agility and reducing costs of business operations. The proposed artifact comes in the form of a process modelling tool. Our proof of concept can be used in both directions – from the physical world to IS and vice versa, thanks to the implemented roundtrip engineering capabilities. Firstly (from physical world to IS), reality can be represented with the help of our modelling tool which brings enhanced analytics capabilities and increased support for operations planning. This way, the information system (IS) can be regarded as being a Digital Twin for the physical environment. Secondly (from IS to physical world), the information system will be able to alter and shape the way in which the physical environment exists. A series of dedicated modelling concepts that describe the actions that need to be done by a drone, allow the users the program the flight scenario and then send it to the drone (physical device) in order to be performed by it.

This sort of coupling between physical and digital world highlights the role of digital engineers and their need for a smart innovation environment, as presented by Karagiannis et al. (2020), where the central stage is occupied by a Conceptual Modelling (CM) workspace. Karagiannis et al. (2020), pointed out that CM will consolidate its presence in digital design and engineering activities, providing a visual representation of the enterprise architecture and a low code solution to enable orchestration of its resources. In correlation to this, our paper addresses the need to enhance the vision regarding the role of IS towards the CM tools, applied on an example from the Industry 4.0 spectrum. In our case, we developed our modelling tool with the purpose of assisting a smart organization that wants to employ drones in business operations in order to gain operational flexibility and competitive advantages.

The work done here is based on a prior version written by the authors (Chis and Ghiran, 2022) in which we presented our proposal and only described a high level overview of the metamodel. In this paper, we proceed in describing the implementation details.

The reminder of this article is structured as follows: Section 2 identifies the problem statement and sets the objectives for the current research, Section 3 describes the required means to enable the operationalization of our work, while section 4 discusses in detail the approach for the research problem, providing important information about the implementation of the modelling tool and its functional components. Section 5 presents the related works followed by a section with the evaluation of the modelling tool in terms of usability and semantic coverage. The article ends with conclusions.

2 Problem Statement and Objective Identification

The concept of Digital Twins has become popular to denote the digital representation of a real system, as presented by Boschert and Rosen (2016). Digital Twins gained a prime role in enabling businesses to optimize workflows and operations, resulting in improved cost effectiveness in business processes, cutting off execution times and reducing errors. This is a consequence of their ability to emulate aspects of businesses before they are implemented, providing a well needed support mechanism for experimentation. Digital Twins allows managers to try different implementation scenarios, leading them to take the best business decisions.

In the literature, according to articles of Kritzinger et al. (2018) and Negri et al. (2017), Digital Twins have been defined as digital representations that are fully integrated with their Physical Twins. In traditional Conceptual Modelling the outcome was Digital Models that represent digital representations of existing physical objects (i.e. they are parallel objects and evolve independently). Then, two directions developed: 1) the model-driven paradigm that empowered Conceptual Modelling with the capability of controlling/guiding the environment and 2) a particular manifestation of the domain specific modelling that explored better representations in the form of Digital Shadows, so that any change in the state of the physical object would change the state in the digital object in an automatic way. The round trip engineering (which we currently embrace in this paper) involves synchronization and maintenance of the models and the elements that were conceptualized in those models (either physical elements or other digital assets). This is an important requirement for ensuring the alignment between the digital and real world and must be kept in mind in identifying the elements that can activate the mediation role of IS.

Therefore, the IS development must definitely consider the development of Digital Twins as a software component that is i) model and data driven (it gets inputs from both cyber and physical objects; i.e., it can be a model that gets input from another model or from some Real World sensors) and ii) can control physical devices (can operate on them). Conceptual Modelling is a "must have" ability for the future Digital Engineer skill profile and it should be treated as a first-class citizen in the development of any enterprise system due to the added value that it brings to the IS.

In this direction, for supporting the Digital Engineer skill profile development, Karagiannis et al. (2020) already pointed out some suggestions regarding what should be considered in operationalizing the additional role of the IS (i.e. that of mediation) that appeared in the last years besides the representation role: on one hand, there are some technological ingredients in the form of metamodelling platforms and on the other hand, there are methodological ingredients in the form of a specific engineering method that should be applied.

This paper presents a CM method meant to emphasize the increasing role that CM is playing in IS and to show the gained value in adopting it against the company's competitors by leveraging a better communication and synchronization between the digital and real world.

The method was developed according to the Design Science Research frame (Wieringa, 2014) (i.e. engineering a method artifact):

Improve *the before mentioned synchronization of the Digital Twins with their physical counterpart;*

By threating it *as a CM method;*

To satisfy *the increasing requirements a company is facing to remain competitive on the market;*

In order to *be aware of the changes in the Industry 4.0 environments (that are reflected in the models) and be able to respond to them or even generate operationalizations (as models can control the environment).*

3 Background

This section describes the means that enable the operationalization of the proposed artifact.

3.1 Technological Enablers

First, the development of IS has been changed towards a low-code/no-code paradigm which means that software developers are expecting to program by simply adding scripting blocks rather than using textual, error-prone (typo) scripts. Model-driven engineering and operation is a handy approach to propagate a change from the digital world to the physical world (it can be also encountered under the label "model-aware engineering", as presented in Buchmann et al. (2018)).

One key enabler is the maturity of metamodelling platforms like ADOxx (ADOxx, 2022), which enables a modelling tool to be generated based on a customized modelling language. The models created with the modelling tool are not simple diagrams, but they benefit of machine reading capabilities given by the metamodel governing them. Treating Conceptual Modelling diagrams as semantic graph structures was demonstrated in the work done by Buchmann and Karagiannis (2016) and made available for experimentation purposes in the Bee-Up tool (Bee-Up, 2022) as an export option of the diagrams in RDF format (RDF W3C Recommendation, 2022).

3.2 Methodological Enablers

A methodology for the engineering process of the IS is expected to manifest an agile and flexible approach, which is further imposed to Conceptual Modelling and its metamodelling design. It requires not only quick adaptability in software engineering, but also for the adopted modelling language (metamodel) and modelling tool (Ambler, 2002).

A metamodel must be defined and evolve in response to the changing needs from the domain it applies to. AMME (Agile Modelling Method Engineering), detailed in the work done by Karagiannis (2022), has been conceived from the beginning to address the development and the change of the metamodel to allow an iterative and incremental way based on an automated prototype generation.

Employing a domain specific Conceptual Modelling method in our analysed business scenario would bring important benefits for stakeholders. These manifest in a modelling method that is tailored to the domain of interest and modellers can use it to better represent the domain (to better simulate it, to analyse it or just describe it). In the same time, due to the close binding between the modelled domain and the real one, functionalities are streamlined, achieving a better coupling between "design-time" and "run-time". Defining a domain specific modelling method starts from a very generic metamodel by adding extensions at conceptual and functional level:

1) Conceptual level extensions refer to i) the addition of new modelling concepts from general ones for which we need to define new attributes and relationships with all the other concepts or ii) to reusing some of the concepts from an existing modelling language for which we have to define some adjustments or iii) to both of the previous ways.

a) Semantic customization is sustained by the enrichment of modelling concepts with new attributes that can provide useful domain specific information to both human and machine actors

b) Syntactic customization is realized by adding or extending the rules regarding the usage of concepts in the modelling tool – ensuring the correctness of all models (e.g. domain and range constraints for relationships, cardinality constraints).

c) Notation level customization is done in order to correlate the meaning of the concepts with their graphical representation, being targeted at human users, aiming to help them employ models in fast way and clearly understand the domain. Moody (2009) highlighted the importance of choosing expressive graphical representations, correlated to the scope of the modelling tool.

2) Functional level extensions regard defining and attaching a series of mechanisms to the modelling concepts, which translate into execution commands – in our case, to control smart devices (e.g. drones) from a work site. This sort of mechanisms can also employ backward functionalities like generating diagrams based on execution scripts (ensuring a better representation in models which is synchronized with the real world) or other functionalities correlated to the scope of the modelling tool.

4 Method Design and Development

Our solution ensures a two-way transition from design-time to execution-time. We implemented roundtrip engineering capabilities allowing diagrams to act as code generators but also in reverse, code can act as diagram generators.

Our developed modelling tool relies on 2 main pillars: 1) incorporating domain specific concepts and 2) integrating functionality that is needed for diagram enactment, roundtrip engineering and drone control. The artifact was built on the ADOxx Metamodelling Platform (ADOxx, 2022), as it enables a fast prototyping environment required by AMME.

4.1 Solution Overview

We wanted to create an aid for implementing drones in Industry 4.0 business operations. If drones were initially employed for aerial video and photography, the current use cases include making deliveries, or supervising large agricultural areas or construction sites (Lidynia et al. 2017).

Drone fleet management has become a requirement for companies that employ them in their operations. Usually, drones come with different control applications and interfaces, depending on the maker and the model. This leads to increased heterogeneity when it comes to managing all the devices, making drone fleet management a complicated process. The modelling tool can support the drones' inventory by providing a visual map of all the enterprise locations with the corresponding drone objects available for each area. Each drone in the model is precisely described using various attributes thus, providing a clear digital representation in the modelling tool. Figure 1 shows an example of this type of model, Drone Resources.

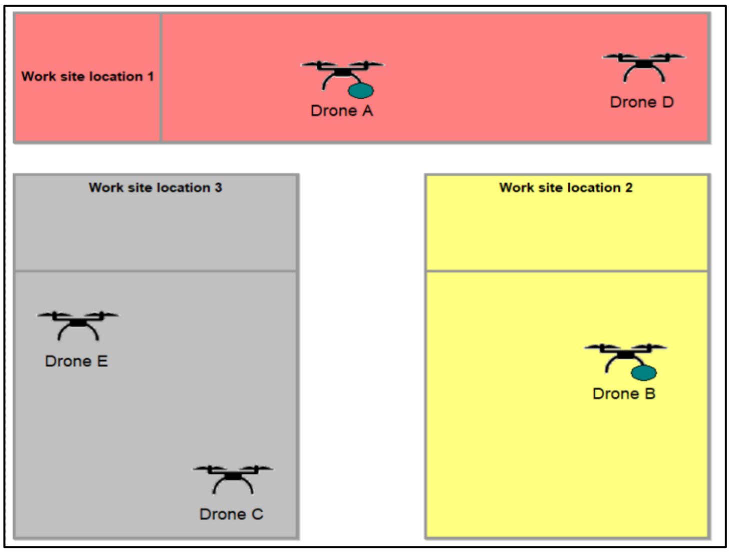

Figure 1. Drone Resources model

But our developed DRONEBPMN modelling tool aims to become more than a representation medium. We would enable all types of drones to be controlled in a model-driven fashion. This automation layer is an instance of ontological reversal because the execution flow is designed and tested in the digital environment, then being enacted in the physical world, adding more agility in operations management inside industrial sectors. Figure 2 represents an example of a model type, called Drone Flight Path that describes the route to be followed by a certain drone.

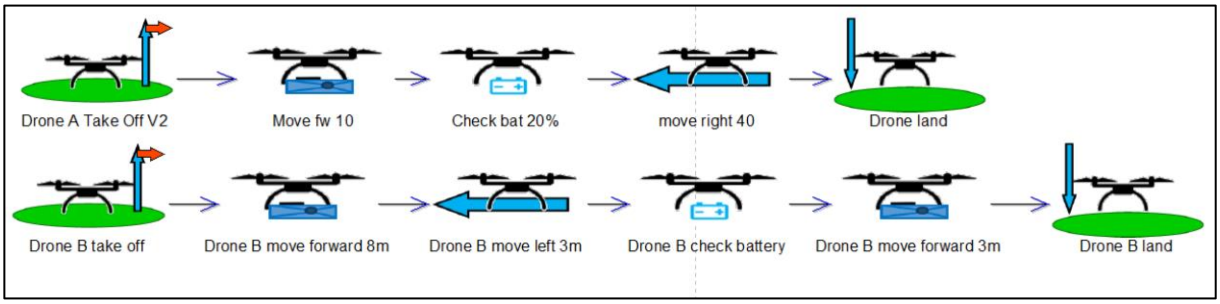

Figure 2. Drone Flight Plan

A task from a business process can be also linked to a drone object integrating the drone control capabilities into the larger business landscape. Companies often rely on business processes for modelling their services and employ business process diagrams to represent the set of activities that aims at accomplishing an organizational objective (Yangui et al. 2021). Business processes have been represented with standardized notations like BPMN (Business Process Model and Notation (BPMN, 2022) and due to the popularity, we kept the existing concepts and properties in the notation and only added few customizations to integrate them with our new concepts.

Figure 3 shows an example of a BPMN diagram in which the process dealing with a work site incident includes the activity "Send Drone A in aerial photo mission". The symbol for this activity is enriched with a visual cue in the top right corner to indicate that the activity is related to another object. The added small circle also acts as a hot spot and modelers can use it to jump to the linked object. We added such linkage functionality in order to support the automation facet of the modelling tool when launching different flight missions. Besides the obvious benefits at functional level, this provides an enhanced coupling between activities and their required operational resources (here, drones) at business level. Managers could benefit from this overview of all the activities and their involved resources in order to have a complete picture of the operations' state, helping them take informed business decisions. Section 6 provides an example of a model query that explores this inter-model semantic links.

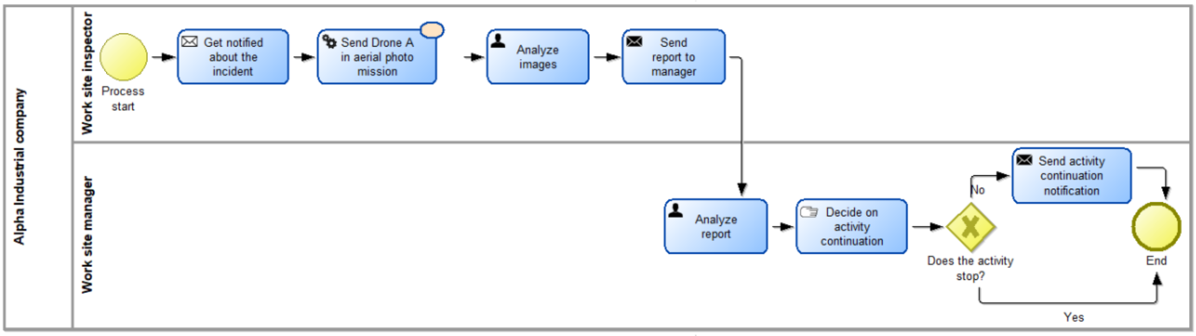

Figure 3. Process of dealing with incidents

A modeller can start by creating Drone Flight Path models, after that can add Drone Resources which will be linked to the previously created flight paths. Each drone can also be linked to a task of a business processes that employ a particular drone (i.e. as a specific resource). The Drone Flight Paths model is parsed and transformed into Python source code which further dictates the operation of drones in real world. Vice versa, drones can send information about their status, which are first captured in Python and then some of the Python scripts can be converted into diagrams to have a visual representation of the drones. Figure 4 describes a bird's view architecture of the tool:

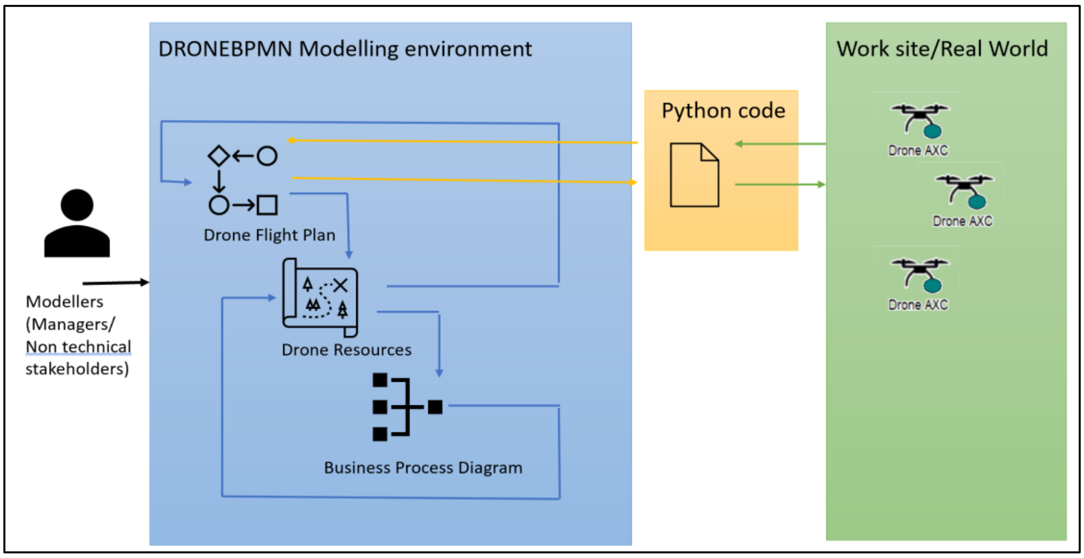

Figure 4. Architectural view of the artefact

4.2 Implementation Details

This subsection describes in detail how we adapted the standard business process modelling notation (BPMN, 2022) to match specific modelling requirements by adding our domain specific concepts and particular functionality.

The engineering process implemented AMME as the go-to methodology when developing modelling concepts and functionalities. We started with the BPMN metamodel on top of which we added a series of new classes, attributes and relationships to support the necessary mechanisms for drone control and roundtrip engineering. The implemented concepts are mapped on the actions that are done by a drone when flying, or to the actual devices. We also implemented a few more abstract concepts that serve in drone fleet management.

Figure 5a) depicts the abstract concepts which are used as more general concepts for inheritance purposes and the model type, Drone Resources. The DeviceNode class (1) serves as a superclass for all the concepts needed to represent smart devices. Under it, we added the DroneCommand (2) concept (also a superclass) and the Drone (3) concept, which belongs to the model type Drone Resources. This model type assists the managers in modelling the location of the drones for a better overview of the drones' positions on the worksites. Hence, it includes another concept called Drone Location (4) which is derived from the BPMN pool class concept.

The Drone class (3) has several attributes: the producer of the drone, the drone's model, what it would be suitable for (e.g. buildings' inspections, aerial photography etc) and the Flight plan attribute. This last attribute is a model link (also called Interref in ADOxx' terminology) to the DroneTakeOff (5) element that will launch the necessary flight mission.

DroneCommand (2) concept is further specialized in other subclasses (6-11) which represent the commands that can be executed by a drone while flying (these are concepts that belong to the model type Drone Flight Path).

As a relation class, we added the nextInSequenceCommand (12) that is used to connect the concepts from the Drone Flight Path model type.

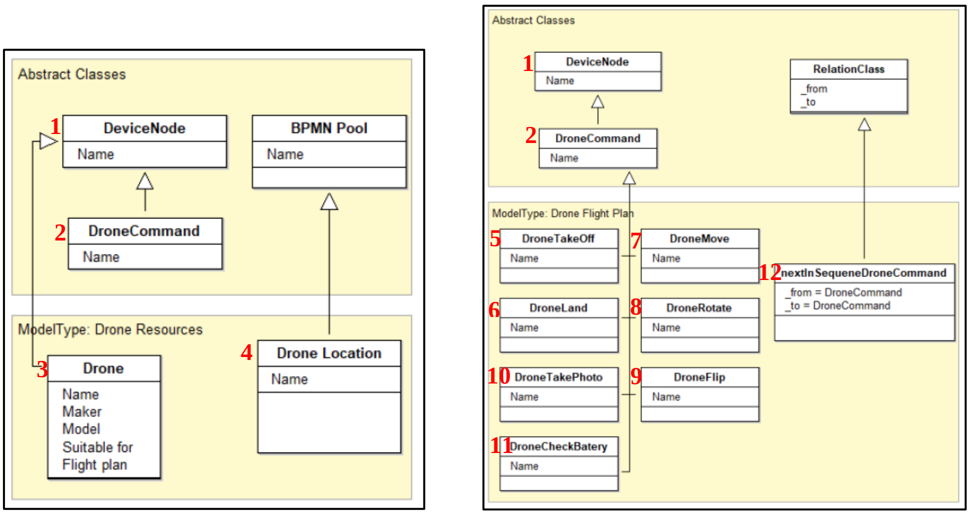

Figure 5. Metamodel for a) Drone Resources model type and b) the Drone Flight Path model type

4.2.1 Semantic Level Customization

We had to enrich the concepts presented in the metamodel from Figure 5, with a series of attributes. This had to be done in order to achieve the technical capabilities required in programming the flight missions in a model-driven approach (machine readability). The added attributes and their usage are detailed further in this section.

The DroneTakeOff (5) has the Execute flight programcall attribute, which calls an AdoScript (AdoScript, 2022) script that generates the flight code in Python. Using the DJITelloPy library (DJITelloPy, 2022), it invokes the Python script that launches the flight.

The DroneLand (6) class doesn't have its own attributes; it only signals the end of a mission and the landing of the drone.

The DroneMove (7) class contains 2 attributes besides the name: the movement direction and the distance in centimetres.

The DroneRotate (8) class has two parameters, the rotation direction (clockwise or counter clockwise) and the number of degrees of the movement.

The DroneFlip (9) class has an attribute which specifies the direction of the flip.

The DroneTakePhoto (10) class has an attribute where the user can specify the path on the computer where the photo obtained from the drone will be saved.

The DroneCheckBattery (11) class has an attribute named "Critical battery level" which is a critical level that needs to be tested before continuing the mission. If the battery level is under the critical level, the drone will land and the mission will be aborted.

The DroneLocation (4) class inherits attributes from the Presentation tab from the BPMN Pool class.

The DroneLocation has only a grouping role, at this point serving more as a visual element for the modeler.

4.2.2 Syntactic Level Customization

There are a few rules that are implemented in the modelling tool in order to ensure the correctness of the models and provide a safe manner of executing the flights. A possible wrong sequence of commands sent to the drone could result in performing a series of dangerous manoeuvres which might threaten the persons or other assets around. In order to avoid this, we designed a series of safety mechanisms which are also transposed in syntactic rules.

In our case, each drone flight model must start with a DroneTakeOff model element and this must be only one over the course of the path. Moreover, each flight ends when encountering a DroneLand concept, which can be placed only once per flight path. By analogy, the DroneTakeOff could be regarded to have the same nature as the Start concept from BPMN, while DroneLand could be seen as the End concept from BPMN.

The DroneTakeOff has no incoming relationships and has only one outgoing, while the DroneLand concept has no outgoing relationships, while having exactly one incoming. All the other concepts from the drone flight diagram have exactly one incoming and one outgoing relationship.

Regarding the model links, the Drone concept from the Drone Resources model type can be only linked the one BPMN diagram at once and also, the Drone can only be linked to one flight path at a time. We imposed a cardinality rule that doesn't allow drones to point to more than one DroneTakeOff concept at a specific time.

4.2.3 Notation Level Customization

We aim to correlate the graphical notation with the meaning of the symbols from our modelling tool, according to the principles for creating modelling notations, described in Moody (2009). As the modelling tool is foreseen to be employed by various stakeholders, friendly and expressive visual notations are of high importance when it comes to increasing the chances of larger scale adoption for DRONEBPMN. The usage of our modelling tool considers the involvement of "citizen development" and "citizen automation" in the sense suggested by Imgrund, 2017.

Figure 6 presents all the modelling notations from our tool, separated by the model type they belong to. One notable feature is the fact that in the Flight Path Model, the symbols are dynamic and they change according to the values of the attributes chosen by the modeler. For example, in the case of a DroneMove, if the moving direction is left, then the arrow will point to the left side.

Similarly, when a concept has an attribute that refers to a link to another element from another model (e.g. a Drone is connected to a flight path, or an activity from a business process is connected to a drone) this is also suggested with a small visual cue on the concept's icon: a small ellipse placed in a corner.

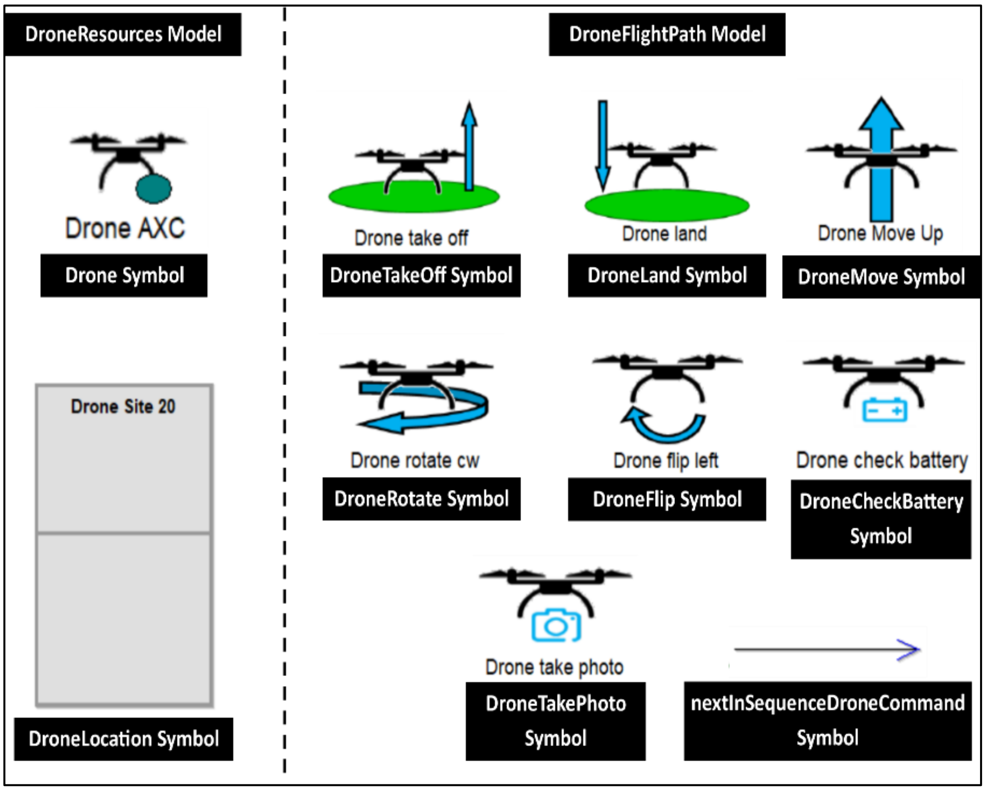

Figure 6. Modelling Notations

4.3 Functional Components

We relied on ADOxx (ADOxx, 2022) with its proprietary scripting language AdoScript (AdoScript, 2022), Python (Python, 2022) and the DJITelloPy library (DJITelloPy, 2022) in order to develop the needed functional components. ADOxx served as the prototyping environment, while AdoScript was used alongside Python to develop the functionalities of the modelling tool. The DJITelloPy library serves in sending commands to the drone, ensuring communication between digital and physical worlds.

Our modelling tool implements two key algorithms – one that parses models and transforms them into executable code and the other one transforms code into a diagram, ensuring the roundtrip engineering facet. Figure 7 displays the key activities of both algorithms.

The models created in the process modelling phase can be regarded as code controllers, dictating the way in which the code is generated and consequently, the way in which the drone will fly.

The DroneTakeOff concept plays the most important role in the flight diagram, as it is used to trigger the diagram parsing algorithm. This generates the Python file that contains all the instructions needed for launching flight missions. The DroneTakeOff concept has an attribute that is represented in the form of a red arrow in the diagram (see figure 2 as an example of a diagram created with this concept), which once it is pressed triggers the parsing algorithm.

The diagram parsing starts when the execution trigger is pressed on the desired flight path, directly in the diagram. After the parsing algorithm is triggered, it will gather the necessary information for the flight, subsequently invoking the Python script generated from the parsed model elements and sending information to the drone.

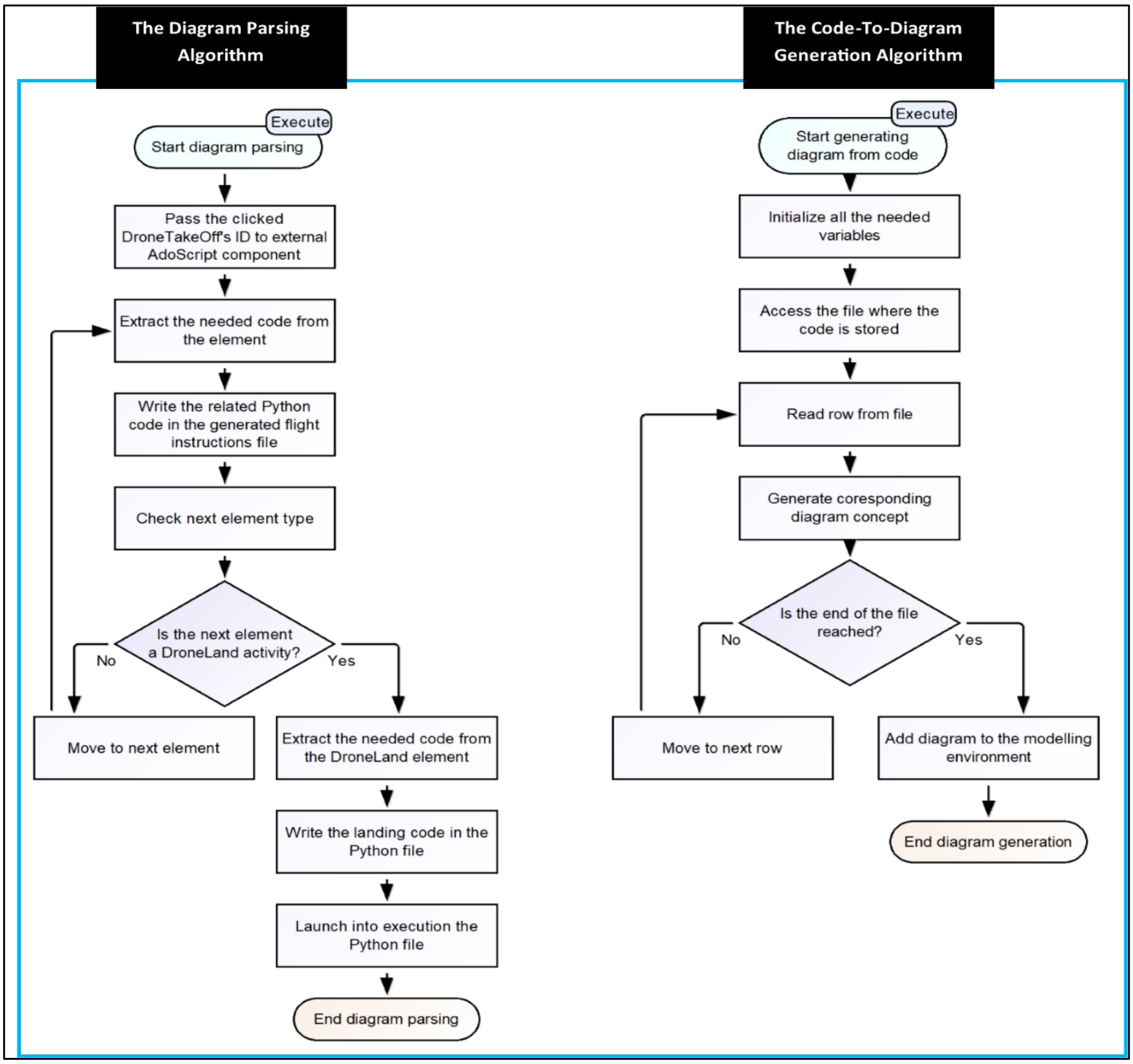

Figure 7. Flowcharts describing the main algorithms implemented in the modelling tool

The generation of a flight diagram from the code is triggered by clicking a menu item in the modelling tool. After all the variables are initialized, the file containing the code is parsed line by line and the written code is transformed into modelling concepts in a roundtrip engineering manner. After the whole file is parsed, the modeler is able to see the generated flight path diagram in the modelling tool.

The runtime code generation mechanism is dependent on the attributes of each modelling concept, which are stored as model content at design-time. As seen in figure 8, the modeler can set the attributes' values which are later on pushed to the run-time part, controlling in this way the flight of the drone. In this case, for a DroneMove concept, we set the name of the concept in the diagram, the movement direction and the flight distance. At run-time, the diagram parsing algorithm will iterate through all these values and they will be pushed after that to the enactment component when the flight is launched from the modelling tool. This approach fully highlights the benefits of model-driven programming, as all these attributes can be modified for the next flight mission, directly in the modelling environment.

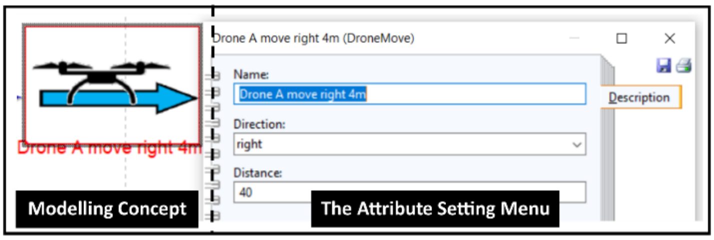

Figure 8. Displaying the modelling concept and the attributes that can be set for it at design time

5 Related Works

We can find many studies regarding one of the two directions: either for applying CM for better representations (from physical world to IS) or in the area of model-driven development, e.g. research regarding the employment of models to dictate code generation, process execution or device control. Our work differentiates from most of the studies as it considers a roundtrip solution.

The work presented by Kirchhof et. al (2022) discusses the creation of MontiThings - a model-driven artifact for generating executable containers which serve for the development of large IoT systems. With this, indeed, developers can focus mainly on the business logic facet and based on that, the models generate the needed code in C++. Our work also allows users to focus on the business rules first and the overall logic of the drone flight, but in our case the generated code is in the Python programming language.

Al-Refai et al. (2014) report on the work done for developing the FiGA framework that allows developers to update code at run-time through changes made to the UML diagrams (UML, 2022). Our work also allows changes in the run-time part to be done seamlessly by modifying the design-part, showcasing a situation of strong coupling model-driven software engineering. Moreover, because of the newly developed modelling language, we are able to capture more domain knowledge from the autonomous drone flights sector.

Similarly, there are many studies in the area of process mining, e.g. generating process models based on captured operational data. Roldán et. al. (2018) applied process mining techniques to analyse and improve multi-robot missions. Their main contribution was regarding the generation of event logs and discovery of models in the context of realistic missions of fire surveillance and extinguishing using a fleet of UAVs (unmanned aerial vehicles).

However, as identified by Brockhoff et al. (2021) the research on process mining and model based device tracking is conducted in different research communities and there is a need to bring them closer. Our paper is meant to further enrich the joint exploration in both areas: we built a demonstrator to highlight the complete roundtrip metamorphosis, even if currently our work is limited in applying any of the existing process mining algorithms and the generated diagrams are obtained by parsing text commands.

Our work is also related to studies regarding the processing of models using machine readable technologies, e.g semantic technologies like RDF (RDF W3C Recommendation, 2022). These enable further processing given by the underlying graph based structure or enable integration with other data sources by adhering to a common representation format. In this sense, a comparable approach could be the work done by Amoui et al. (2012). It describes the development of the GRAF (Graph-based Runtime Adaptation Framework) that employs a model-centric approach to managing graph-based models of software at runtime. This allows increased flexibility in changing the run-time functionalities of a system due to graphs' ability to be dynamically modified. Even though our approach is based on models rather than graphs, by using the features of ADOxx, we can export our diagrammatic models as RDF graphs which can be queried by SPARQL language (SPARQL W3C Recommendation, 2022). Otherwise, developers or modelling practitioners can stick to the diagram-centric approach that generates the needed code for enacting the drone's flight process.

6 Evaluation

This section focuses on evaluating the DRONEBPMN modelling tool, in terms of semantic coverage and usability. Currently, our evaluation is limited to feasibility assessment as we primarily wanted to point to the two main facilitators that empower the CM to become the essential tooling to any enterprise IS. Considering the Framework for Evaluation in Design Science Research proposed by (Venable et. Al, 2016), first we need to set the goals of the evaluation, which were articulated by the previous statement. Then, regarding the evaluation strategy we adopted the validation through a proof-of-concept. Our example served as a demonstrator of the applicability of the technological (metamodelling platforms) and methodological (AMME modelling method methodology) enablers to pave the way for the new generation of modelling tools agnostic to the domain of interest. In terms of the properties to evaluate, we checked the possibilities of incorporating domain specific concepts (for a better representation of the business specificity) and of integrating functionality (required for the diagram enactment). The developed modelling tool (created using the specified means) proved an ease in addressing the before mentioned characteristics both in the development stage of the modelling tool but also in its operation phase.

6.1 Semantic Coverage

This evaluation criteria is meant to address the degree to which a CM mediator (in the form of a modelling tool) can be integrated in the specific realm for each enterprise. Incorporating domain specific concepts is seen as a model value creation (Buchmann, 2022), making the created models more purposeful as they adequately employ which concepts or properties should be captured to achieve that purpose. Our DRONEBPMN tool addressed this requirement by providing concepts that are domain specific and more, these new concepts could capture the semantics pertaining to each domain in the form of machine readable and specific properties. Nowadays, without the possibility to represent process-related data and various event data (many of these originating in the digital environment) or limiting the representation to simple PowerPoint/drawings diagrams, managers miss important insights for the success of their company (also called "tactical and strategic intelligence" (Polyvyanyy et. Al 2017)). The possibility to query the content of the models is a must for the modern Digital Entrepreneur. We only show here, for demonstration purposes some simple queries that could be executed in the modelling tool, however, more advanced queries, that can combine information outside the modelling tool are enabled if we export the diagrams in RDF as we have already mentioned the existence of this option in the Bee-Up tool.

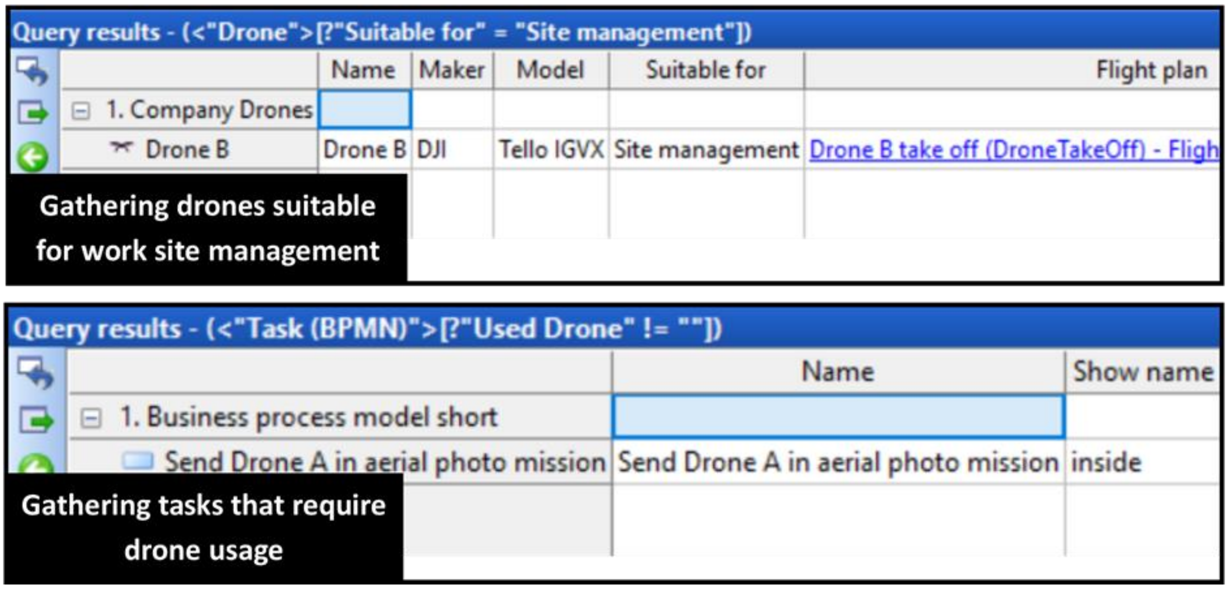

Figure 9. AQL Queries in the DRONEBPMN Tool

AQL Queries (AQL Queries, 2022) available in ADOxx Modelling and Bee-Up allow to perform query-based analysis of the DRONEBPMN models. Figure 9 exemplifies two AQL queries.

The first query will gather all the drones that are suitable for work site management from our DroneResources model, alongside all their attributes:

(<"Drone">[?"Suitable for" = "Site management"]),

the result being visible in the upper side of figure 9.

The second query will get all the tasks that require the usage of a drone, from the BPMN model:

(<"Task (BPMN)">[?"Used Drone" != ""]),

the result being visible in the lower side of figure 9.

6.2 Usability Evaluation

This evaluation criteria assessed the manner in which the added functionality (both model driven process execution and diagram generation) could be efficiently carried out from the modelling tool. A usability evaluation made here reports on the number of clicks that a modeler needs to perform in order to use the DRONEBPMN modelling tools, as seen in Table 1.

Operation	Number of Clicks
Linking BPMN task to flight path – DRONEBPMN modelling tool	15
Launching the drone mission (starting from the BPMN model) – DRONEBPMN modelling tool	3

Table 1. Usability evaluation metrics for the modelling tool

The results presented in the table show the fact that modelers and stakeholders can perform all the semantic links easily with our modelling tool, which greatly improves the way in which processes can be managed across multiple situations.

7 Conclusions

Through this paper, we reframed Conceptual Modelling as an enabler for the more prominent role (that of mediation besides representation) that an information system is playing in the enterprises' mission in the current, Digital First era. The paper is an extension of our previous work, providing a concreate example of a modelling tool that is a proof-of-concept for the model-driven flavour of Digital Twins. The DRONEBPMN modelling tool aims to close the gap between digital representation of Industry 4.0 processes and their physical enactment. Our proof-of-concept tool comes as an important aid in integrating the usage of drones with industrial processes, which can significantly reduce costs, execution times and also human resource risks on industrial sites. We managed to create a bridge between digital and physical environments, providing a roundtrip engineering approach for creating, deploying and testing processes that use drones in correlation with industrial business activities. We managed to achieve the strongest coupling in model-driven software engineering, thanks to the bidirectional propagation of changes. The modelling tool generates flight code for the drone, but also changes in code will be seen in the models, which is an important aid in managing business processes.

Our future works will continue the research started in the domain of coupling the design-time and the run-time facets of business processes, extending our tool to more diverse scenarios and allowing the integration of a broaden plethora of smart devices and robots.

References

AdoScript (2022). *The AdoScript Scripting Language*. URL: https://www.adoxx.org/live/scripting-language-adoscript (visited on 15 November 2022).

ADOxx (2022). *The ADOxx Metamodelling Platform*. URL: https://www.adoxx.org/live/home (visited on 13 November 2022).

Al-Refai, M., Cazzola, W. and France, R. (2014). “Using models to dynamically refactor runtime code”, in: Proceedings of the 29th *Annual ACM Symposium on Applied Computing*, Gyeongju, Republic of Korea.

Ambler, S. (2002). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. Wiley.

Amoui, M., Derakhshanmanesh, M., Ebert, J. and Tahvildari, L. (2012). “Achieving dynamic adaptation via management and interpretation of runtime models”, *Journal of Systems and Software*, 85 (12), 2720-2737.

AQL Queries (2022). *Query Language AQL*. URL : https://www.adoxx.org/live/adoxx-query-language-aql (visited on 10 November, 2022).

Baskerville, R.L., Myers, M.D. and Yoo, Y. (2022). “Digital first: The ontological reversal and new challenges for is research”, *MIS Quarterly*, 44 (2), 509-523.

Bee-Up (2022). *The Bee-Up Modelling Tool*. URL: https://bee-up.omilab.org/activities/bee-up/ (visited on 13 November 2022).

Boschert, S. and Rosen, R. (2016). “Digital twin—the simulation aspect”, *Mechatronic Futures*, 59-74.

BPMN (2022). *The Business Process Modelling and Notation Standard*. URL: https://www.bpmn.org/ (visited on 9 November 2022).

Brockhoff, T., Heithoff, M., Koren, I., Michael, J., Pfeiffer, J., Rumpe, B., Uysal, M.S., Van Der Aalst, W.M. and Wortmann, A. (2021). “Process prediction with digital twins”, in: *ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion* (MODELS-C), Fukuoka, Japan.

Buchmann, R. A. and Karagiannis, D. (2016). "Enriching Linked Data with Semantics from Domain-Specific Diagrammatic Models," *Business & Information Systems Engineering*, 58 (5), 341-353.

Buchmann, R. A., Cinpoeru, M., Harkai, A. and Karagiannis, D. (2018). “Model-aware software engineering - A knowledge-based approach to model-driven software engineering”, in: Proceedings of *ENASE* 2018, Funchal, Madeira, Portugal.

Buchmann, R.A. (2022). “The Purpose-Specificity Framework for Domain-Specific Conceptual Modeling”. In *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools* (pp. 67-92). Cham: Springer International Publishing.

Chis, A., Ghiran, A.M. and Buchmann, R.A. (2022). "Proof of Concept for a Roundtrip Engineering IS for the New Enterprise in the Industry 4.0 Era", Proceedings of the 24th International *Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, Linz, Austria

DJITelloPy (2022). *The DJITelloPy Python Library Reference*. URL: https://djitellopy.readthedocs.io/en/latest/ (visited on 12 November 2022).

Imgrund, F., Fischer, M., Janiesch, C. & Winkelmann, A. (2017). “Managing the Long Tail of Business Processes”, *ECIS* 2017 Proceedings, Guimaraes, Portugal, pp. 595–610.

Karagiannis, D. (2022). “Conceptual Modelling Methods: The AMME Agile Engineering Approach”, *Domain –Specific Conceptual Modeling*, 3-21.

Karagiannis, D., Buchmann, R.A., Boucher, X., Cavalieri, S., Florea, A., Kiritsis, D., and Lee, M. (2020). “OMiLAB: a smart innovation environment for digital engineers”, in: Camarinha-Matos, L., M., Afsarmanesh, M., & Ortiz, A. (eds) *Working Conference on Virtual Enterprises*, Valencia, Spain.

Kirchhof, J.C., Rumpe, B., Schmalzing, D. and Wortmann, A. (2022). “MontiThings: Model-driven Development and Deployment of Reliable IoT Applications”, *Journal of Systems and Software*, 183, Article 111087.

Kritzinger, W., Karner, M., Traar, G., Henjes, J. and Sihn, W. (2018). “Digital Twin in manufacturing: A categorical literature review and classification”, *IFAC-PapersOnLine*, 51 (11), 1016-1022.

Lidynia, C., Philipsen, R. and Ziefle, M. (2017). “Droning on about drones—acceptance of and perceived barriers to drones in civil usage contexts”. In *Advances in Human Factors in Robots and Unmanned Systems*: Proceedings of the AHFE 2016 International Conference on Human Factors in Robots and Unmanned Systems, July 27-31, 2016, Walt Disney World®, Florida, USA (pp. 317-329). Springer International Publishing.

Moody, D. (2009). “The “Physics” of notations: towards a scientific basis for constructing visual notations in software engineering”, *IEEE Transactions on Software Engineering*, 35 (5), 756–779.

Negri, E., Fumagalli, L. and Macchi, M. (2017). “A review of the roles of digital twin in CPS-based production systems”, *Procedia Manufacturing*, 11, 939-948.

Polyvyanyy, A., Ouyang, C., Barros, A. and van der Aalst, W.M. (2017). Process querying: Enabling business intelligence through query-based process analytics. *Decision Support Systems*, 100, pp.41-56.

Python (2022). *The Python Programming Language.* URL: https://www.python.org/ (visited on 12 November 2022).

RDF W3C Recommendation (2014). *RDF Resource Description Framework 1.1 Concepts and Abstract Syntax*. URL: https://www.w3.org/TR/rdf11-concepts/ (visited on 10 November, 2022).

Roldán, J.J., Olivares-Méndez, M.A., del Cerro, J. and Barrientos, A. (2018). “Analyzing and improving multi-robot missions by using process mining”, *Autonomous Robots*, 42 (6), 1187-1205.

SPARQL W3C Recommendation (2013). *SPARQL 1.1 Query Language*, URL: https://www.w3.org/TR/sparql11-query/ (visited on 11 November, 2022).

UML (2022). *The Unified Modeling Language*. URL: https://www.uml.org/ (visited on 1 November 2022).

Venable, J., Pries-Heje, J. and Baskerville, R., (2016). “FEDS: a framework for evaluation in design science research”. *European journal of information systems*, 25, pp.77-89.

Wieringa, R.J., (2014). *Design science methodology for information systems and software engineering*. Springer.

Yangui, S., Goscinski, A., Drira, K., Tari, Z. and Benslimane, D. (2021). “Future generation of service-oriented computing systems”. *Future Generation Computer Systems*, 118, pp.252-256.