Towards ontology enabled agent-based twinning for cyber-physical systems

Reference:
Marah Hussein, Albertins de Lima Lucas, Challenger Moharram, Vangheluwe Hans.- Towards ontology enabled agent-based twinning for cyber-physical systems
2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), 1-6 October, 2023, Västerås, Sweden - ISBN 979-83-503-2498-3 - IEEE, (2023), p. 444-448
Full text (Publisher's DOI): https://doi.org/10.1109/MODELS-C59198.2023.00078
To cite this reference: https://hdl.handle.net/10067/2019190151162165141

Towards Ontology Enabled Agent-based Twinning for Cyber-physical Systems

Hussein Marah*, Lucas Lima*†, Moharram Challenger*, Hans Vangheluwe*
*Department of Computer Science, University of Antwerp & Flanders Make Strategic Research Center, Antwerp, Belgium
{hussein.marah,moharram.challenger,hans.vangheluwe}@uantwerpen.be
†Departamento de Computação, Universidade Federal Rural de Pernambuco, Recife, Brazil
lucas.albertins@ufrpe.br

***Abstract*—The twinning paradigm is a concept that embraces end-to-end amalgamation and bridging the physical and cyber worlds. Harnessing the power of twinning to cyber-physical systems offers several benefits and opportunities, that are encountered by multiple challenges. Data integration, knowledge representation, reasoning, and inferencing are crucial for providing dependable services to the twinned systems. To address these challenges, an ontology-enabled approach for modelling cyber-physical systems and their twins is proposed, which is incorporated with an intelligent agent-driven representation of the different elements of the system. Applied to a case study, this approach enables representing the knowledge of the system and its environment, capturing different aspects and integrating heterogeneous data from various sources. This allows performing reasoning and inferencing on the system to obtain valuable insights and information.**

***Index Terms*—Ontology, Agent-based Approach, Twinning Paradigm, Cyber-physical Systems**

I. Introduction

The manufacturing industry is encountering several challenges and complications (modelling the knowledge of the system and the environment as it evolves through time, handling data from heterogeneous components and sources, providing dynamicity and adaptability, defining autonomous behaviours, and managing heterogeneous components and distributed decision-making of Cyber-Physical System (CPS)) following the massive integration of CPS and Internet of Things (IoT) technologies within different systems to achieve automation and smart operations [1].

The twinning paradigm or Digital Twin (DT) [2] refers to a virtual representation of a real-world process or system, closely mirroring its physical counterpart from multi-perspectives in terms of properties, data and behaviour. Obviously, DTs would offer several benefits to model and build smart systems utilising the interaction between the physical and digital worlds that establishes a dependable connection to integrate industrial information effectively [3]. Accordingly, designing and implementing DT for a complex CPS in a dynamic environment requires modelling their components on different levels of abstraction. For instance, physical systems of DT should be provided with autonomous capabilities to react and adapt to changes in the dynamic environment. This task can be accomplished using an agent-based approach [4] emphasising the individual entities, or agents, within a system. These agents possess features and capabilities such as autonomy, intelligence, pro-activity, and the ability to interact with others to achieve their goals. On a more abstract level, DTs should be able to carry on more intensive, intelligent and complicated operations that support and provide valuable services to the system [5].

The agent-based approach supports modelling CPS and their DTs from different perspectives and at different layers of abstractions. The agent-based approach can capture this heterogeneity by modelling DTs for each component as an independent agent, accommodating variations in capabilities and decision-making processes of each component. Thus, DTs can be scaled up or down depending on complexity. Ultimately, the agent-based approach has been leveraged to model and build DTs for CPS. Monitoring, anomaly detection, and other services can be quickly built and integrated with agents in the CPS components and their DTs, utilizing the flexibility and scalability of this architecture. Despite all the features powered by agent-based DTs, they still lack managing the knowledge of the different components in the system, inferring and reasoning on this knowledge on demand to acquire specific information, and defining the system components on different abstraction levels to perform intensive and complicated tasks that require detailed contexts and deep overview of the system's status.

As stated in the literature, ontology is a facilitator for providing domain knowledge of independent components of CPS [6], [7], bridging the real-world knowledge to a digital space. Some of the works, such as [8], utilize ontology to cover the fundamental concepts necessary for mapping physical devices and developing DTs, [9] present ontology-based implementation to enhance decision-making between collaborative entities, and [7] suggest modelling the behaviour of the production system as a semantic data model. However, [10] reports that there is insufficient attention to the twin data (components knowledge) and its internal evolutionary mechanisms as separate entities. Accordingly, agents can handle the autonomous, distributive and individual encapsulation of DTs and CPS elements. At the same time, ontology can efficiently represent, maintain and update the knowledge of these elements in a repository where they can be easily retrieved.

Ontology-driven modelling methods help to represent the systems' knowledge by incorporating them into ontology, which defines the concepts and relationships within the target domain of the System under Study (SuS). The modelling process becomes more structured and standardised. Ontology provides a shared vocabulary and a formal representation of knowledge, enabling effective communication and interoperability between the different actors within a single system and other systems. The integration of ontology offers powerful querying, analysis, and reasoning capabilities. Specific states, situations, and system details can be easily accessed and analysed using ontological reasoning. This facilitates the extraction of valuable insights and supports decision-making processes.

By leveraging ontology capabilities and an agent-based approach, this paper aims to establish an advancement in modelling DTs for complex systems. Thus, it contributes to developing more robust, flexible, inter-operable, and knowledge-rich DTs deployment. This work has shown promising results of combining ontology and the agent-based approach to model the knowledge and different aspects of CPS. This provides an effective and practical approach for modelling, designing, and implementing intelligent DTs of complex systems.

The remainder of the paper is structured as follows. Section II describes our proposal for digital twin systems employing ontologies and agent-based systems. Section III shows how we apply our approach in the context of a multi-robot system for warehouse automation. Finally, Section IV presents our final remarks and future directions.

II. The Proposed Approach

The proposed approach leverages the power of ontologies coupled with the agent-based DTs deployment [11], [12]. This way, the system represents CPS entities distributively as autonomous agents with their capabilities and behaviours. Ontology representation can capture and model complex relationships, semantics, and domain knowledge. This enables us to gain a contextual understanding of the system's behaviour, interactions, and dynamics in the environment, which might comprise a set of other operational and related systems, as envisioned in Fig. 1.

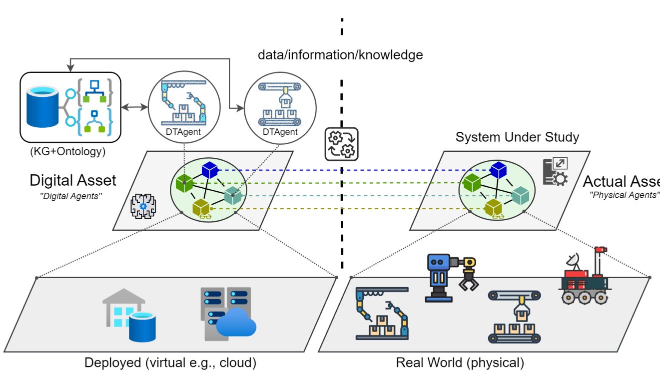

Fig. 1. The Generic Overview of the Ontology Enabled Agent-based DTs.

The architecture describes the realization of the ontology-integrated agent-based digital twins, which comprise twofold and is exemplified in Fig. 2.

A. Agent-based DTs

The box on the right side in the architecture given in Fig. 2 is the `Agent-based Digital Twins` deployment, which is the actual implementation of DTs of a particular CPS elements (e.g., sensors and actuators of the robot) powered by agents as in the multi-robot case study. This system consists of multiple mobile robots modelled as autonomous agents (`Physical Agents`) in the `Physical Agents Layer` to collect information, operate and control (movement, orientation and navigation) the system. Then, DTs agents (`Digital Agents`) are virtualized and twinned for each component of the implemented CPS to provide the `Digital Twin Services` in the `Digital Agents Layer`.

B. Ontology Integration

The proposed architecture benefits from ontology and leverages its concretisation (`Knowledge Graph`) as elucidated in Fig. 2, which describes the deployment of the second part. The ontology is designed, defined and instantiated considering the system's requirements and identifies precisely which parts of system knowledge should be modelled. It might contain knowledge of the system components or other processes and states or the environment knowledge, including its characteristics, conditions and dynamics.

We have used Ontology Modeling Language (OML) and its IDE (Rosetta[1]) to define ontologies. From the OML vocabularies, we can derive the Web Ontology Language (OWL)/Resource Description Framework (RDF) specifications that are loaded in a *Fuseki Server*[2]. Fuseki is part of Apache Jena, a Java framework for building applications based on semantic web technologies like RDF and OWL. With Fuseki, it is possible to deploy a server for storing graph structures based on RDF triplestores. The graph can be queried using SPARQL[3], which is a W3C standardised declarative language to query RDF using a notation similar to SQL. SPARQL provides statements not only to retrieve data but also to insert or delete data in the RDF graph, enabling updating the knowledge of agents-based DTs. Using Fuseki, we can expose SPARQL endpoints in order to allow external applications to execute SPARQL statements for updating or querying the RDF graph. Fuseki also has a built-in Web Graphical User Interface (GUI) that enables users to perform SPARQL statements.

Therefore, in our architecture, we use the OWL/RDF serialisation to represent the knowledge graph that follows the ontology defined in OML. This graph is materialised in the Fuseki server, and we expose SPARQL endpoints to allow interaction with the knowledge graph. This interaction is orchestrated by a REST API that exposes services for the `Agent-based Digital Twins` to interact with the knowledge graph. The

[1]https://github.com/opencaesar/oml-rosetta
[2]https://jena.apache.org/documentation/fuseki2/
[3]https://www.w3.org/TR/rdf-sparql-query/

exposed services provide capabilities for both keeping the knowledge graph up-to-date with the digital twin, but also, to perform reasoning on the data to support the digital twin's decision-making. Lastly, the web GUI provided by Fuseki can also be used to interact with the knowledge graph, for instance, to execute some query that is not yet available as a service. Nevertheless, the user must own skills to manipulate SPARQL and be aware of the concepts and relationships of the ontology.

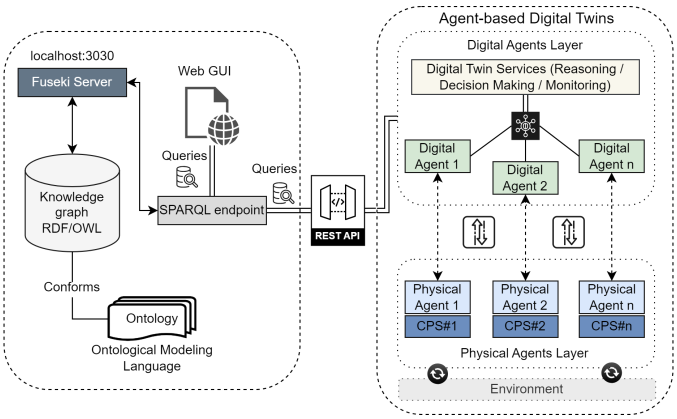

Fig. 2. The System Architecture of Ontology-integrated Agent-based DTs.

III. Case Study & Scenarios

This section presents a specific case study to illustrate our exploration of using ontologies for agent-based twinning.

A. Case Study

Warehouse logistics refers to an automated multi-robot warehouse that shifts from traditional manual labour to more automated methods using mobile robots, which are self-driving vehicles that carry products, packages and materials within the warehouse [13].

Implementing multi-autonomous robots within a warehouse environment offers several benefits by improving processes, such as task allocation, picking, sorting, and transportation. However, designing and building such a complex system and modelling the dynamics of the environment where the robots operate puts several challenges and requirements (safety, knowledge representation, managing operations, data integration, power management and handling scalability) that should be addressed and satisfied. Thus, an effective modelling approach is needed to design, model and manage the knowledge and data of the CPS components and the environment. Knowledge and data are used to analyse, measure, and monitor the system's behaviour to take the most appropriate actions.

In order to establish this system, the proposed approach is used to model the agent-based part, and the ontology deployment is also realised accordingly. Thus, the agent-based DTs implementations (i.e., the box on the right of Fig. 2) of the warehouse system are provided by utilizing the framework presented in [12]. That being the case, the DTs of the multi-robot warehouse is deployed by agents (using the JADE[4] MAS platform) and ready to be used but without any ontological reasoning capabilities.

Integrating ontology into such a bare-bone system enriches and adds several advantages in the context of knowledge representation, managing data from different sources, and inferring the knowledge to support reliable decision-making of this system. For this reason, the realisation of the ontology (i.e., the box on the left of Fig. 2) is considered using the stack of technologies presented in Section II. Fig. 3 depicts a simplified graphical representation of the developed ontology in OML. Here we have concepts to represent the warehouse digital twin, its physical environment and layout, and the robots that drive through this layout. The robots move according to their movement task, which indicates a motion through a line of the layout from one vertex to the other. Other types of tasks may exist, for instance, picking a package, but they are not depicted here for simplicity.

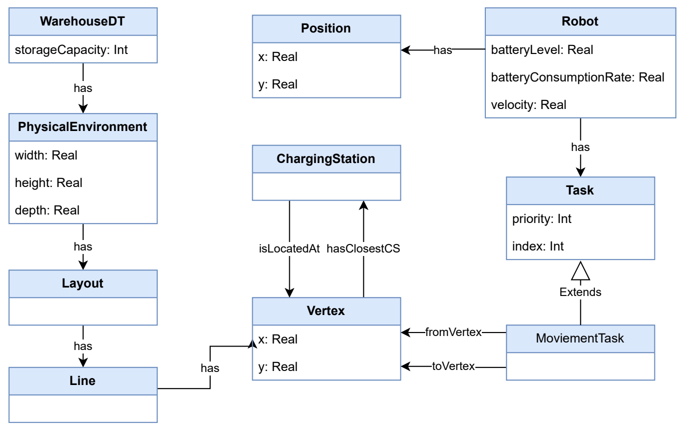

Fig. 3. Partial Ontology Representation for the Warehouse Case Study.

Once the ontology is developed and validated in Rosetta, we generate the OWL/RDF files to be loaded into the Fuseki server to materialise our Knowledge Graph (KG). Once the `Agent-based Digital Twins` part is running, it uses the REST API to feed the KG with data. Likewise, it reasons and performs inference over this data.

The end of this step results in an ontology-integrated agent-based DTs of the target system. The following subsection is dedicated to presenting some real-case scenarios of the benefits of this integration and how it would favourably add powerful features and capabilities to the system.

B. Multi-Robot Warehouse Scenarios

The studied and inspected scenarios are given in the context of the multi-robot warehouse system. The warehouse's environment and the setup are sketched in Fig. 4. Simply, a determined map of the warehouse environment contains lines and vertices (A, B, etc.), which are connected to define the trajectories the robots should follow. The vertices represent either a package destination (picking or delivery) or a charging station. The warehouse is equipped with ultra-wideband technology for indoor localization in the warehouse system. Thus,

[4]https://jade.tilab.com/

the trajectories mentioned earlier are virtually determined in the warehouse. Basically, missions and movement tasks are distributed and assigned to various robots through the warehouse control system. Every individual robot can have several movement tasks, and every single movement task is represented by one directed line between two vertices. The robot can start from any location along the movement task between two vertices. For example, it can be denoted as $\vec{AB}$, which connects an initial point A with a terminal point B and vice versa, as illustrated in Fig. 4.

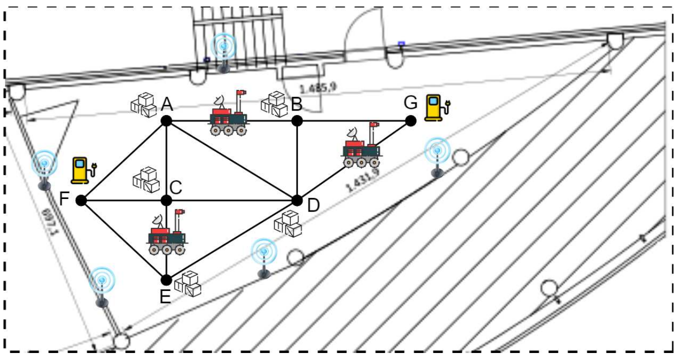

Fig. 4. The Schematic Overview of the Warehouse Map.

a) Collision Detection Scenario: Collision detection is essential for maintaining the safe and effective mobility of the robots throughout the warehouse. The main goal of collision detection is to prevent robots from crashing into one another or other objects, reducing the possibility of downtime, damage, and potential safety risks.

Predicting a collision usually relies on incorporating extra external sensors, such as onboard visual systems or RGB-D cameras strategically positioned in the surroundings [14]. Many methods and algorithms have been developed to implement collision-free robots. Yet, based on the screened literature, ontology has not been considered to model and address this problem. In this scenario, we show how the robots and the environment are modelled using ontology (KG) and integrated into the agent-based DTs of the warehouse. Every robot agent updates its corresponding instance's status (e.g., position, speed, etc.) in the KG. Based on the queried situation, the agent gets information about possible collisions by inferring the available knowledge.

Collision detection has been deployed with ontology by identifying and calculating some of the properties of the robot and the environment. We list some possible collision scenarios: (1) Find the Estimated Time of Arrival (ETA) of every robot that drives to the same vertex (located on different lines) to detect a possible collision at that specific vertex; (2) Identify a possible collision between two robots driving to the same vertex but at different speeds. In this case, the faster robot will collide with the slow robot's rear part; (3) Detect a collision between robots located on the same line but driving in opposite directions towards each other. Listing 1 illustrates a concrete example of a SPARQL query used to calculate ETA for the first collision scenario.

In principle, robots have different velocities, and they operate in the environment in different locations as they perform different tasks. The KG is updated accordingly based on the new information and the environment dynamics. Thus, this query calculates the *ETA* by considering the robot's velocity, current position and the distance to reach a particular intersection point where a collision might happen.

```
1  PREFIX rob: <http://ua.be/warehouserobots/vocabulary/robot#>
2  PREFIX base: <http://ua.be/warehouserobots/vocabulary/base#>
3  PREFIX wh: <http://ua.be/warehouserobots/vocabulary/warehouse#>
4  PREFIX afn: <http://jena.apache.org/ARQ/function#>
5  SELECT ?robotID ?robV ?dis ?eta WHERE {
6    ?robot a rob:Robot .
7    ?robot base:hasIdentifier ?robotID .
8    ?robot rob:hasCurrentPosition ?pos .
9    ?robot rob:hasVelocity ?robV .
10   ?pos rob:hasX ?robX .
11   ?pos rob:hasY ?robY .
12   ?robot rob:hasTask ?task .
13   ?task rob:hasIndex 0 .
14   ?task rob:fromVertex ?fromV .
15   ?task rob:toVertex ?toV .
16   ?fromV base:hasIdentifier ?fromVID .
17   ?toV base:hasIdentifier ?toVID .
18   ?toV wh:hasX ?vX .
19   ?toV wh:hasY ?vY .
20   bind((?vX-?robX)*(?vX-?robX) as ?subx)
21   bind((?vY-?robY)*(?vY-?robY) as ?suby)
22   bind(afn:sqrt(?subx+?suby) as ?dis)
23   bind(?dis/?robV as ?eta)
24 }
```

Listing 1. Finding the Estimated Time of Arrival (ETA) of Every Robot.

Lines 1-4 define prefixes that are used in the query to make it more concise. Line 5 specifies what is being returned, in this case, the robot identifier, the robot velocity, the calculated distance to the vertex, and the *ETA*. Lines 6-19 determine the patterns to be found in the KG. For instance, Line 6 defines a variable *?robot* that must be typed by *rob:Robot* (see Fig. 3). Using these patterns and mapping elements to variables, we can access the relevant data in the KG. For calculating the *ETA*, we need the current position of the robot (*?pos*), its velocity (*?robV*), and the position of the vertex the robot is going (*?toV*). Then, from lines 20-23, we perform the calculations for the *ETA*, binding the results to variables.

TABLE I
RESULTS OF THE ETA QUERY.

robotID	robV	dis	eta
r1	14.0	1328.0	94.85
r2	24.0	2500.0	104.16
r3	28.0	772.0	27.57
r4	20.0	1078.0	53.90
r5	20.0	1078.0	53.90

Table I shows the results of conducting such a query for five robots. The query shows information about every robot, the robot ID (*robotID*), its velocity (*robV*), the distance to reach a particular destination (*dis*) based on that situation, and finally the estimated time of arrival to that destination (*eta*). However, this query is part of a more extensive query that calculates each robot's *ETA* and compares the ones that aim to reach the same vertex. Considering a threshold determined to anticipate the future closeness of every robot to each of the other robots that drive towards the same destination, we can return the ones that will probably collide at that point based on the difference of the calculated *ETA*.

Collision detection includes several queries, and multiple factors are considered. Agent-based DTs can use the REST API to update the shared knowledge in the KG instances and seamlessly access and retrieve the information by executing queries autonomously and collaboratively with other agents making the most appropriate decision in such a dynamic and uncertain environment to avoid the collision.

b) Power Efficient Scenario: The lack of efficient power management systems and methods for mobile robots in large dynamic/unknown environments limits their usage and applications [15]. Therefore, robust and long-haul power management solutions in mobile robots are required. For this reason, users can benefit from using our proposed approach to model and design energy-efficient mobile robots.

The goal of this scenario is to enable the robots to autonomously navigate in their environment and effectively estimate their battery consumption for a particular movement task and then select the closest charging station and most efficient route from their location if the battery level is insufficient to complete the task. This is performed by inferring the information from the current knowledge of the robots and the environment. The steps considered in this scenario are enumerated as the following, (1) Calculate the Estimated Battery Consumption (EBC) for the final destination if the robot has multiple movement tasks; (2) Compute the *EBC* for every line (the whole path) and calculate the *EBC* for the closest charging station.

First, we identify the *EBC*, which depends on how much power the battery the robot consumes every second. Since we are using a LEGO Raspberry Pi-BrickPi[5], as an estimation, a robot powered by fresh AA batteries may have a consumption rate of around 1-2% per minute during active operation. Thus, we assume the battery consumption rate to be 0.025% per second. This leads to multiplying *ETA* by the consumption rate (0.025%) in order to compute the *EBC* of a robot's mission.

Finally, a query is executed to find the difference between the current battery level and the calculated *EBC* to decide whether the robot can reach the destination ($if\ EBC <= RobotBatteryLevel \rightarrow Continue$) or must charge its battery in the nearest battery station ($if\ EBC > RobotBatteryLevel \rightarrow Charge$). Note that these concepts, like *batteryLevel* and *batteryConsumptionRate*, are available in the ontology. This allows us to reason on the desired property, in this case, the *EBC*.

IV. Conclusion

In conclusion, this paper has explored the concept of enabling CPS with agent-based DTs integrated with ontologies. Integrating ontological knowledge and the agent-based approach has shown significant potential in enhancing the capabilities and effectiveness of DTs in the context of CPS and its environment. By leveraging ontologies, which formally represent domain knowledge, such as in the implemented case study of the multi-robot warehouse, DTs of the robots can better understand and reason about the system they represent, leading to improved decision-making, system performance and utilisation of resources in the warehouse.

Through the deployment of agents, DTs become intelligent entities capable of autonomously interacting with the physical world through their representative physical agents, acquiring and processing data, and making decisions by inferring and reasoning on existing and updated knowledge. The ontology-enabled agent-based DTs show a good potential to design complex CPS where the system is comprised of heterogeneous distributed components, and the knowledge of the system comes from different sources. We plan to explore scenarios like this in the future using federated KG, including the concerns with scalability and performance of the whole system.

References

[1] A. Corallo, V. Del Vecchio, M. Lezzi, and P. Morciano, "Shop floor digital twin in smart manufacturing: a systematic literature review," *Sustainability*, vol. 13, no. 23, p. 12987, 2021.

[2] M. Grieves and J. Vickers, "Origins of the digital twin concept," *Florida Institute of Technology*, vol. 8, pp. 3–20, 2016.

[3] L. Li, B. Lei, and C. Mao, "Digital twin in smart manufacturing," *Journal of Industrial Information Integration*, vol. 26, p. 100289, 2022.

[4] U. Pakdeetrakulwong, P. Wongthongtham, W. V. Siricharoen, and N. Khan, "An ontology-based multi-agent system for active software engineering ontology," *Mobile Networks and Applications*, vol. 21, pp. 65–88, 2016.

[5] E. Karaarslan, Ö. Aydın, Ü. Çalı, and M. Challenger, "Digital twin driven intelligent systems and emerging metaverse," 2023.

[6] M. Garetti, L. Fumagalli, and E. Negri, "Role of ontologies for cps implementation in manufacturing," *Management and Production Engineering Review*, 2015.

[7] E. Negri, L. Fumagalli, C. Cimino, and M. Macchi, "Fmu-supported simulation for cps digital twin," *Procedia manufacturing*, vol. 28, pp. 201–206, 2019.

[8] C. Steinmetz, A. Rettberg, F. G. C. Ribeiro, G. Schroeder, and C. E. Pereira, "Internet of things ontology for digital twin in cyber physical systems," in *2018 VIII Brazilian symposium on computing systems engineering (SBESC)*, pp. 154–159, IEEE, 2018.

[9] M. Rico, M. L. Taverna, M. R. Galli, and M. L. Caliusco, "Context-aware representation of digital twins' data: The ontology network role," *Computers in Industry*, vol. 146, p. 103856, 2023.

[10] Z. Ren, J. Shi, and M. Imran, "Data evolution governance for ontology-based digital twin product lifecycle management," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 1791–1802, 2022.

[11] H. Marah and M. Challenger, "An architecture for intelligent agent-based digital twin for cyber-physical systems," in *Digital Twin Driven Intelligent Systems and Emerging Metaverse*, pp. 65–99, Springer, 2023.

[12] H. Marah and M. Challenger, "Madtwin: a framework for multi-agent digital twin development: smart warehouse case study," *Annals of Mathematics and Artificial Intelligence 2023*, pp. 1–31, 7 2023.

[13] L. Custodio and R. Machado, "Flexible automated warehouse: a literature review and an innovative framework," *The International Journal of Advanced Manufacturing Technology*, vol. 106, pp. 533–558, 2020.

[14] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292–1312, 2017.

[15] M. U. Farooq, A. Eizad, and H.-K. Bae, "Power solutions for autonomous mobile robots: A survey," *Robotics and Autonomous Systems*, vol. 159, p. 104285, 2023.

[5] https://www.dexterindustries.com/brickpi