A multi-robot warehouse system : an exemplar

Reference:
Marah Hussein, Paredis Randy, Challenger Moharram, Vangheluwe Hans.- A multi-robot warehouse system : an exemplar
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. 530-538
Full text (Publisher's DOI): https://doi.org/10.1109/MODELS-C59198.2023.00090
To cite this reference: https://hdl.handle.net/10067/2019200151162165141

A Multi-Robot Warehouse System: An Exemplar

Hussein Marah, Randy Paredis, Moharram Challenger, Hans Vangheluwe
Department of Computer Science, University of Antwerp & Flanders Make Strategic Research Center
Antwerp, Belgium
{first name.last name}@uantwerpen.be

***Abstract*—The complexity of modelling and designing cyber-physical systems arises due to their heterogeneous components and the need to explicitly model the features of interest for these components from different points of view. To support the modelling of these complex systems, multi-paradigm modelling advocates explicitly modelling every part of a system at the most appropriate level(s) of abstraction. From this perspective, this exemplar paper presents an exemplary application of multi-paradigm modelling on the multi-robot warehouse system to provide architectural, behavioural and operational models. These models demonstrate how multi-paradigm modelling has been applied to model the different system parts at different levels of abstraction, applying different modelling formalisms and notations. This paper aims to provide a reference repository of the collection of models of the multi-robot warehouse case study, which can be accessed and utilized to replicate this system, conduct more future research on this case study, and extend it using other formalisms and modelling approaches.**

***Index Terms*—Multi-Paradigm Modelling, Cyber-Physical Systems, Multi-Robot Warehouse, Modelling Formalisms.**

I. Introduction

Cyber-Physical Systems (CPS) are systems that integrate computation, networking, and physical processes. CPS seamlessly combines hardware and software resources, which are co-designed with the physical components. To engineer CPS, it is necessary to use physical, computational, and communication models [1]. The heterogeneity of CPS modelling poses challenges, including defining modelling languages, ensuring model determinism, and accurately representing causally related discrete events occurring simultaneously.

In addition, engineering CPS requires a combination of methods and approaches from various domains like mechanical and electrical, etc., along with computer science methods. This need emphasizes the importance of trans-disciplinary modelling approaches that integrate different engineering disciplines.

Multi-Paradigm Modeling (MPM) [2] advocates explicitly modelling all aspects of a system at their most appropriate level(s) of abstraction, utilizing appropriate modelling formalisms to address the heterogeneity and multidisciplinary encountered during the engineering of complex systems.

The rise of online shopping and the exponential growth in online retail has dramatically transformed the logistics industry, presenting unusual challenges and opportunities. To meet the increasing demands for efficient order processing, warehouses are considering moving from traditional manual labour to automated systems, with the promising potential of intelligent Multi-robot Warehouse systems, which is a facility that utilizes multiple robots to perform various tasks in the warehouse. These advanced facilities leverage the capability of robotics and intelligent systems to optimize operations, efficient processing, and improve productivity [3].

Deploying multiple robots [4] working collaboratively within a warehouse environment offers numerous advantages. Moreover, by leveraging the capabilities of each robot and orchestrating their actions, multi-robot systems can significantly enhance different tasks in warehouses, from allocating tasks for different robots and performing other tasks such as picking, packing, sorting, and transporting packages [5].

A Multi-robot Warehouse is an example of a complex CPS. As their potential benefits become increasingly evident, the design and engineering of their models, including the models of the high-level warehouse system [3] and the individual robots [6], have drawn significant interest from academia and industry alike. Therefore, this exemplar paper is dedicated to showing how an MPM4CPS approach was used to model and design the Multi-robot Warehouse system and provides various system models, presenting a comprehensive overview of their underlying architectures, functionalities, and operational dynamics.

This paper provides an exemplary case study of an MPM Multi-robot Warehouse. The primary objective of this paper is to provide a structured overview of the different models and modelling formalisms that have been considered for the design, implementation, and execution of the different aspects of the Multi-robot Warehouse system. We explain the main models used in the warehouse system, including structural, functional, operational, and behavioural models, such as the internal warehouse architecture, communication model, navigation strategy, and physics and kinematics model of the robots. By presenting a range of models and their respective features, this paper aims to provide practical reference and reusable artefacts to researchers, experimenters, and practitioners for designing and implementing Multi-robot Warehouse systems.

II. Related Work

A warehouse or a shop floor, representing a high-level unit of manufacturing systems, maintains various types of valuable resources, models, and production-related actions, playing a central role in executing the essential system responsibilities. Based on the literature, multiple works have been conducted that focus on modelling, designing, and building warehouse or shop floor systems for managing robots and production processes. Thus, several models, formalism, and notations have been used to realize these solutions.

Utilizing Unified Modeling Language (UML), the paper [7] presents the concept of a digital twin in the context of material flows for forecasting and observation purposes. UML is employed to represent the system's structure and its connections. The authors of [8] introduced an approach for building a digital workshop replica using Model-Based Systems Engineering (MBSE). They created a structured semantic representation of the workshop system using the Systems Modeling Language (SysML). A multi-scale modelling framework is introduced [9] for facilitating the development of models, starting from the individual component level, progressing to the system level, and extending to the system-of-systems level. The proposed framework uses SysML in the model fusion for machines. An approach for creating a Shop-floor digital twin (SDT) using a meta-model was described in this work [10]. A meta-model is developed based on RAMI 4.0. illustrating manufacturing resources and their states. Finally, the paper [11] presents an architecture and modelling approach [12] for intelligent agent-based digital twins for cyber-physical systems. Thus, this study paves the way for exploiting software agents combined with MPM for developing collaborative digital twins representing different properties of interest.

Following the concepts of model-driven software engineering (MDE), the work [13] discusses the application of MDE in developing heterogeneous multi-robot systems. The authors introduce a framework that utilizes basic actions as the foundation for creating more intricate tasks. Two main models are defined in the framework, the robot model represents the robot itself, and the task model represents the tasks the robots can perform. Another paper [14] introduces an Industry 4.0 approach for modelling and simulating flexible production based on MDE. It presents a Domain-Specific Modelling Language implementation called MultiProLan, for creating production process models that can be automatically enriched or manually designed to support dynamic resource management, production documentation, and error handling.

Based on the principles of Multi-agent Systems (MAS), some studies are modelling different behavioural aspects of multi-robot systems. For example, the collaboration of robots via MAS in a multi-robot system is elaborated in [15] and [16]. The agent-based modelling and implementation of multi-robot systems considering their localization and path planning is studied in [17] and [18]. And finally, in [19], the application of agent-based multi-robot system in a smart production line is investigated.

This paper aims to introduce an exemplary application of the multi-modeling approach to model, design, and deploy a multi-robot warehouse system. The paper explains the combinations of different models and formalisms resulting in an operational system. Applying MPM at this step allows for identifying possible and open gaps for improvements and considering diverse notations and multi-formalism modelling.

III. Background

A. Multi-Paradigm Modelling (MPM)

Multi-Paradigm Modelling (MPM) [2] advocates explicitly modelling every part of a system at the most appropriate level(s) of abstraction, using the most appropriate modelling language(s), view(s) and workflow(s). It focuses on how multiple formalisms can be combined and used in conjunction such that the benefits of all these languages are explored.

The warehouse use-case can be seen as a multi-component system, in which each component is designed, maintained, and given semantics via other models (in different formalisms). Instead of having a translation onto the same common denominator language [20], [21], a hybrid language [22], an embedding of languages [23], or co-simulation [24], the combination of formalisms and models happens in this use-case due to the agent-based nature of the system. Of course, there are high-level interactions between all components, but they are, for the most part, *disjoint* (a.k.a. *loosely coupled*), allowing much more distinction between formalisms. A downside of this approach is that the interfaces need to be well-defined, such that communication between components is possible. Furthermore, this approach is made possible because each component functions on its own and does not necessarily rely on the presence or absence of these components.

B. Systems Modeling Language

Systems Modeling Language (SysML)[1] is a versatile modelling language explicitly designed for systems engineering. It serves as a comprehensive tool for specifying, analyzing, designing, validating and capturing various aspects of systems, including complex systems. SysML extends a subset of the Unified Modeling Language (UML), incorporating additional features and notations to enhance the representation of systems engineering.

A Multi-robot Warehouse is a type of system-of-systems (SoS) that can be significantly complex. A SoS refers to a group of specialized or task-oriented systems that combine their resources and capabilities to form a more sophisticated system and collaborate to accomplish a single objective. This collaborative facility allows the resulting system to provide advanced functionality and enhanced performance than the combination of its individual constituent systems. A Multi-robot Warehouse, for example, comprises numerous autonomous robots and interconnected subsystems, a warehouse management system, a positioning system, and control software. Each individual robot within the Multi-robot Warehouse can be seen as an individual system itself (software and hardware integration) with its own sensors, actuators, decision-making capabilities, and communication abilities. These individual robots collaborate and interact with each other, with their digital representatives (in the case of digital twins), and with other subsystems to perform tasks within the warehouse.

Therefore, the modelling and designing part of this system have been done using SysML, which can model the different

[1] https://www.omg.org/spec/SysML/

facets of the system, the physical and the digital, their architecture, relations and behaviours. SysML offers various diagram types to represent different aspects of a system, such as:

1) Block Definition Diagrams (BDD): Represent the structural aspects of a system, including the various components (blocks) and their relationships.
2) Internal Block Diagrams (IBD): Represent the system components, entities, and connectors as blocks to depict the relationships and interfaces between these components.
3) Activity Diagrams (ACT): Describe the behaviour of the system by representing activities, actions, decisions, and control flows.
4) State Machine Diagrams: Capture the state-based behaviour of components by representing states, transitions, and events.
5) Requirement Diagrams: Specify and trace requirements throughout the system development process.

However, despite its broad functionality, SysML by itself does not have defined operational semantics (i.e., the executive behaviour of a system) other than that interpreted by a human or a specific tool. Therefore, SysML by itself does not suffice for using MPM to create a fully functional warehouse; hence, other modelling languages are also required.

C. *Causal-Block Diagrams (CBDs)*

A common methodology for visual modelling languages is the usage of a (potentially hierarchically composed) "box-and-arrows" notation. This is used in a plethora of modelling languages and domain-specific languages [25]. Causal-Block Diagrams (CBDs) [26] specializes this notation: arrows denote signals (through time), and the boxes denote operations that happen on the values of these signals. For instance, two signals with respective values 3 and 4, entering a summation box, will result in an output signal of 7 (because $3 + 4 = 7$).

Typical for CBDs is their one-to-one relationship with Ordinary Differential Equations (ODEs), which allows for any set of mathematical equations to be converted from ODE to CBD and vice-versa.

Furthermore, CBDs have a time domain of $\mathbb{R}$. This implies that at every point in time, each signal has a value. Therefore, time-based equations (e.g., using integrals and derivatives), self-describing equations and iterative root-finding algorithms can be modelled in CBDs. Because computers are not able to compute the simulation at an infinite precision, a time-based discretization is made and a zero-order hold is assumed between two computation points. This discretization can either be at fixed points in time (e.g., every 1 second), or it can use algorithms like Runge-Kutta Fehlberg of the fourth order (RKF45) [27] to make this step-size *adaptive* (or *varying*).

D. *Multi-Agent Systems (MAS)*

A Multi-Agent System (MAS) is a distributed system consisting of a group of agents that interact with each other and the environment to achieve their individual goals and, in some cases, collective and common objectives [28]. MAS is designed based on the agent-based approach.

In MAS, each agent is an autonomous entity with its perception, decision-making capabilities, and actions. Agents can have different characteristics, behaviours, and knowledge, which may evolve through time due to the dynamic and changing environment.

MAS can be utilized to design distributed, modular, flexible, adaptable and scalable systems with agents that possess autonomous, reactive, proactive and interactive features, providing the system with powerful capabilities.

IV. Multi-robot Warehouse Formalisms

The process of designing, modelling and implementing the Multi-robot Warehouse has considered different models, formalisms and perspectives. Thus, we divided these aspects into two main categories: (1) design and architecture focus and (2) implementation and deployment focus, and each category considers the relevant models from the warehouse perspective and robots perspective as well.

Fig 1 shows the methodology to obtain the Multi-robot Warehouse. This is given as a Process Model (PM; a.k.a. a workflow model), following the extended Formalism Transformation Graph and Process Model (FTG+PM, [29], [30]) notation. The blue, bold arrows show the control flow between activities (rountangles). This flow can be split/merged (blue rectangles) throughout execution. The green, slim arrows show the data flow and link all artifacts (green rectangles) that are produced/consumed by activities. Multiple ports can be used to maintain a clear overview. The upside-down fork icon in the activities indicates that these are hierarchically decomposable into other workflows but are hidden (i.e., collapsed) for more readability/variability. This section will refer to the activities of this workflow to show the link to the models.

A. *Design and Architecture Focus*

Constructing and designing the robots and the warehouse happened in a model-based approach, allowing for *replicability* [31] of this system setup. This section details the modelling of the two main parts of the Multi-robot Warehouse: (1) the robots and (2) the warehouse system and the environment.

During the `RequirementElicitation` activity, we try to understand and formulate the purpose of the warehouse system and its specific requirements (e.g., order fulfillment speed, storage capacity, product types). We also determine the number of robots needed, how the robots should (logically) behave, the size of the warehouse, and the layout. The key requirements that should be satisfied by the warehouse system:

1) R1: Task Allocation: The system should be able to allocate tasks to multiple robots based on their availability, battery level, and proximity to the task location.
2) R2: Path Planning: The system should reduce the duration of missions by planning and utilizing efficient routes.
3) R3: Order Fulfillment: The system should supervise the movement of robots as they transport objects from one

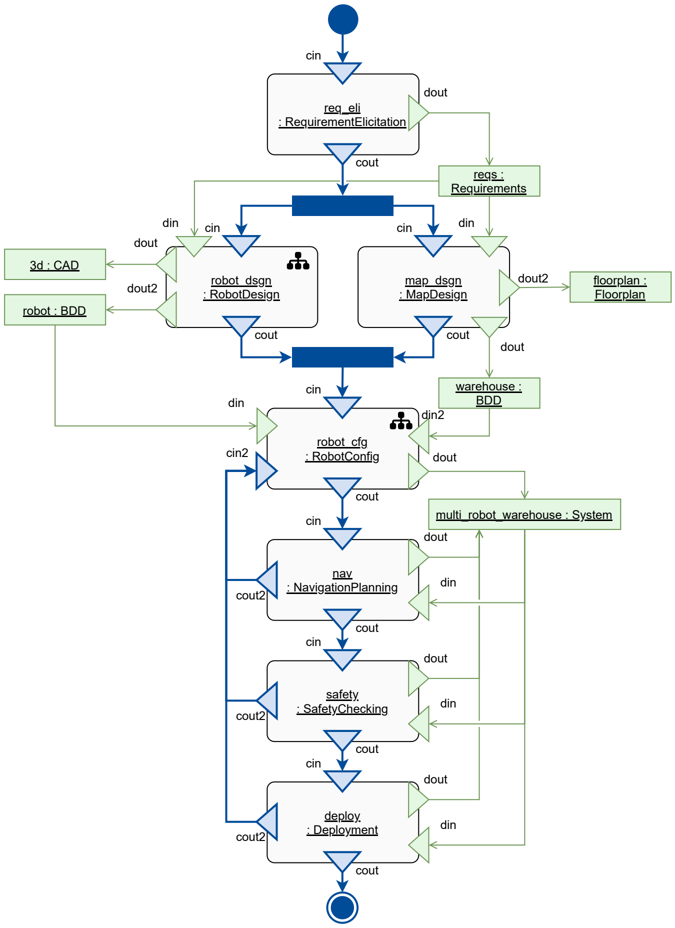

Fig. 1. A workflow model for constructing the warehouse system.

location to another, potentially involving various and sequenced paths.

4) R4: System Monitoring: The system should monitor the status of robots, tasks, and overall system performance to identify any issues.
5) R5: Safety Navigation: The system should be able to prevent collisions between the robots, and the robot should be able to identify the encountered obstacles and safely deal with these situations.
6) R6: Scalability: The system should be able to handle an increasing number of new robots and new tasks.
7) R7: Power management: The system should optimize the power consumption of the robots and ensure the robots complete their assigned tasks efficiently.

1) Warehouse

The warehouse system models include the models of the main components, such as the environment and control system, that construct the whole system by still neglecting parts of the robots' models. According to the requirements and objectives of the warehouse system, the environment can be defined and modelled in the control system. Modelling such systems and their environment depends on several factors and criteria that should be considered in the requirements.

a) Warehouse Floorplan

The model of the warehouse floorplan (obtained from the `MapDesign` activity) is provided as a plain schematic diagram which represents the elements of the environment and parts of the system (e.g., sensors and storage, charging stations) using abstract graphic symbols as exemplified in Fig. 2. Most of the main components are provided in this diagram. The model shows an example of the warehouse setup, where five wireless transmitters of the Ultra-wideband (UWB) anchors (UWB refers to wireless communication technology for positioning and localization that utilizes a wide spectrum of frequencies to transmit data over short distances sensors and uses a very low energy level for short-range) are installed and mounted in the warehouse wall and calibrated to provide the required positioning for the moving robots. On the other hand, using these anchors, the UWB sensors in the robots will communicate with the spread anchors through the warehouse to identify their real-time position.

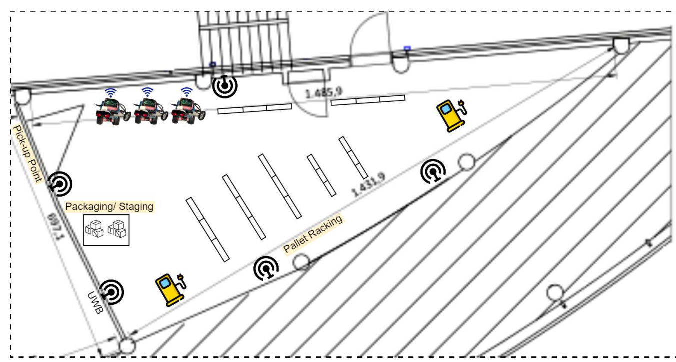

Fig. 2. Warehouse floorplan schematic model.

In the warehouse setup, a *pick-up* point determines the point from which the packages arrive and are then processed in a temporary station in the *packaging and staging*. Robots can be assigned to perform missions from that point. If a robot accepts a mission, it means it is busy executing that mission. The delivery point could be virtually any point on one of the *pallet racking* places in the warehouse.

The floorplan model is programmatically modelled in the warehouse system. This enables the warehouse system to control the operations and assign tasks to the robots. Also, by using visual and graphical representation in the system, users can control, manage and observe the operations of the robots. This floorplan is a dynamic description of the used warehouse setup, which can be easily modified and changed.

b) Warehouse SysML Models

This part explains the provided models in SysML that can be accessed from an open-source repository[5]. These models describe the structure of the entire warehouse system.

Warehouse BDD: Fig. 3 exemplifies the model of the warehouse system (also obtained from the `MapDesign` activity). Note that the warehouse comprises a control system, robots, a map (layout) and the optional deployment of a digital twin, which will not be considered in this paper. The `Control`

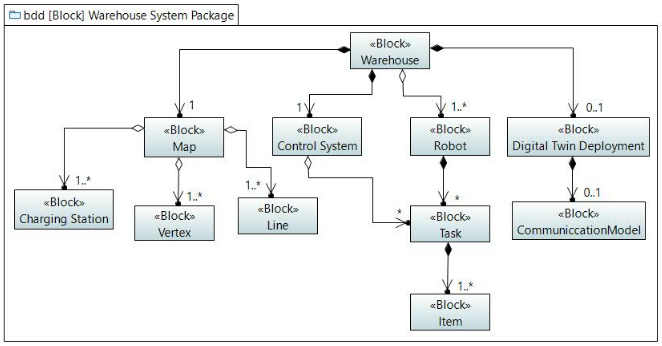

Fig. 3. Block definition diagram of the warehouse system.

`System` is the central unit that manages all the operations. Task allocation, monitoring and alerts are handled through this system. The `Map` is the layout of the warehouse system, where it contains lines, vertices and charging stations that form the trajectories the robots should follow to source to target destinations. It is important to clarify that the map can be physical (lines physically determined in the map), or they can be virtually determined by the localization system. Finally, the robots, indeed, are the main components of the system. The warehouse system can host a minimum of one `Robot`, or as many as the system can handle, which is primarily related to the scalability characteristic of the system.

2) Robots

The robots are simple, nonholonomic differential-drive robots, as described in [32] and [30]. This implies there are two wheels driven by different motors, yet oriented on the same axis. Running both motors with the same amount of power makes the robot move linearly (forwards or backwards), and a different power will help the robot do turns. They are built initially to follow a line drawn on the ground but have been extended to the current use-case. As a consequence, there is a colour sensor at the front to identify the colour of the line. Fig. 4 shows a free-body diagram of this setup.

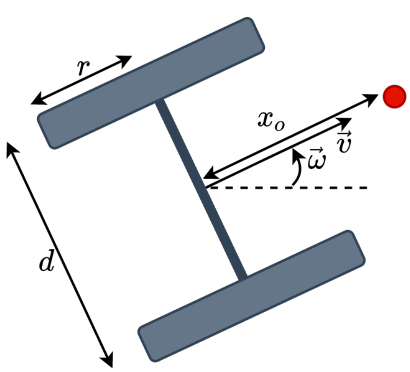

Fig. 4. Free-body diagram of the robots. The circle identifies the position of the line sensor. The axle length is indicated with d, the wheel radius with r, the sensor offset with x_o and its linear and angular velocities, respectively, with $\vec{v}$ and $\vec{\omega}$.

a) CAD Model and Weight Model

From this setup and the notion that the robots are built using *LEGO MINDSTORMS EV3* (313131[2]), we can construct a CAD model, as shown in Fig 5, and an LDraw model[3] for the building instructions. This CAD model is loaded in Blender[4], allowing weight distribution analysis, resulting in a semi-accurate calibration of the center of mass.

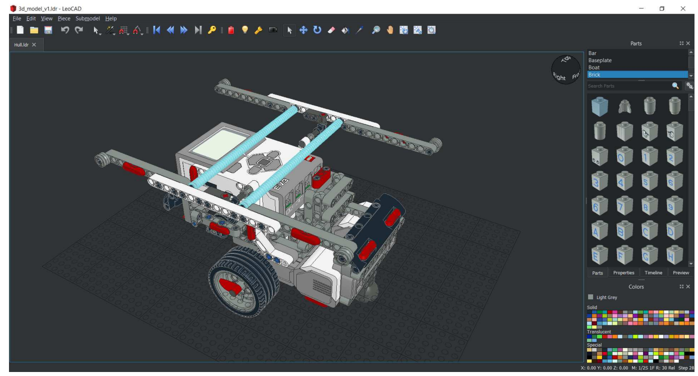

Fig. 5. CAD model of a possible AGV.

b) Robot SysML Models

Using the SysML formalism, structural description and internal components of the robot, the relationship with other components has been modelled in the BDD, IBD and ACT diagrams (obtained from the `RobotDesign` activity). Those models also can be accessed from an open source repository[5].

Robot BDD: The robot is constructed from multiple physical and software components and elements. Physical components are exemplified in Fig. 6, such as `Sensor`, `Actuator`, and `Part`. For example, different types of sensors, such as `Ultra-Wide Band`, `Color Sensor`, and `Ultra-sonic Sensor`, are used to build the robot. On the other hand, the motors that move the robots' wheels `Servo Motor` are an example of an `Actuator`. The `Part` is any other essential component in the robot, such as `Battery` and the `Chassis` of the robot.

The software parts are, basically, the blocks that represent the models used to control and operate the physical and hardware components. The two examples of these are the `Kinematic Model` represented in `Differential Drive Model` used for movement control, and the `Geometric Model` represented in the `Point-to-Point Model` used for navigation.

Robot IBD: It is essential to clarify how the blocks of the robot system are structured, the working mechanism, and how they interact and influence each other. The IBD of the robot provided in Fig. 7, elucidates the relation between the different blocks and how they influence each other. Essentially, the robot uses its sensors (e.g., UWB sensors for positioning and an ultra-sonic sensor for discovering objects) to perceive and

[2]https://www.lego.com/nl-be/product/lego-mindstorms-ev3-31313
[3]https://www.ldraw.org/
[4]https://www.blender.org/
[5]https://github.com/husseinmarah/multi-robot-warehouse-models

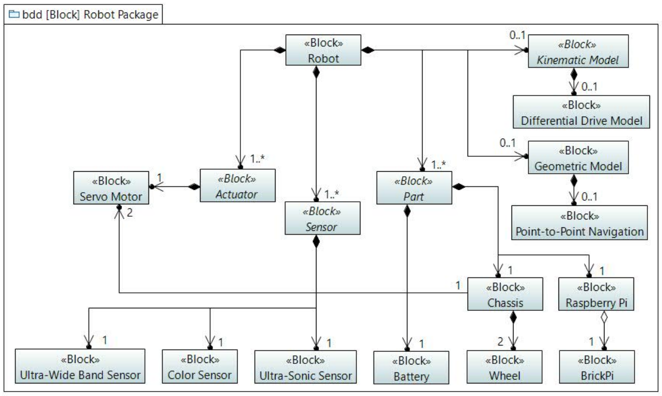

Fig. 6. Block definition diagram of the robot.

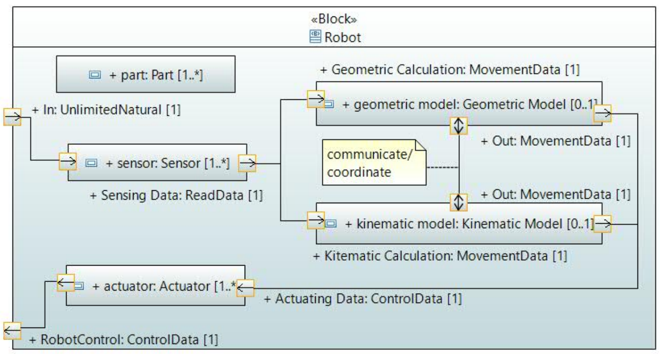

Fig. 7. Internal block diagram of the robot.

sense the changes in the environment. Then, this information is used by models responsible for movement and orientation (e.g., the Kinematic model and Geometric model) to give the proper instructions to enable the robot to navigate smoothly and safely.

Robot ACT: The robot in a warehouse should be able to move and carry objects once it receives a task from the warehouse control system. In such a case, the warehouse control system sends the location of the target destination (waypoint). Then, the robot will be able to calculate the distance and adjust its direction towards the target location. Simply, the point-to-point navigation model, as shown in Fig. 8, is used so the robots can move autonomously in the environment, detect obstacles and avoid collisions. Of course, in the case of obstacles or collisions, robots should re-plan their paths to reach their destinations with the help of the kinematics model of the robot.

The robot kinematics model, which is responsible for moving the robot itself, is described by a set of ODE equations, which will be given in the next section.

B. Implementation and Deployment Focus

From the models obtained in the design focus, it is easy to apply model transformations to gain the models that focus on the executive behaviour of the system. This is done in the

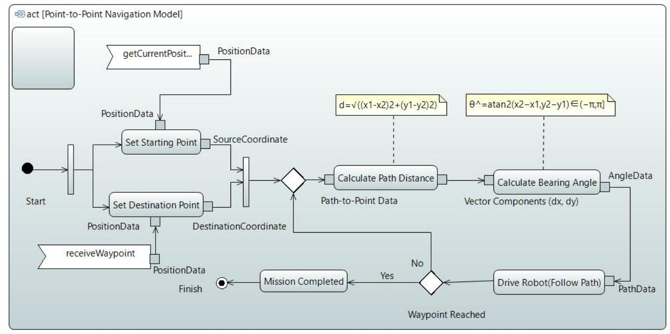

Fig. 8. Activity diagram of the point-to-point navigation model of the robot.

hierarchical `RobotConfig` activity and allows for multiple iterations at later points in time.

1) Robots

The robot's behaviour is described by a set of equations that represent its movement.

a) Ordinary Differential Equations (ODE)

[33] describes the (simplified) detailed set of equations that allow the robot to move forward, including the relationship between motor power and linear/angular velocities. These equations are as follows (ϕ_L and ϕ_R are the left and right wheel speeds, respectively):

$$\dot{\phi}_L = \frac{1}{r} \cdot \left(\dot{v} - \frac{\dot{\omega} \cdot d}{2} \right)$$
$$\dot{\phi}_R = \frac{1}{r} \cdot \left(\dot{v} + \frac{\dot{\omega} \cdot d}{2} \right)$$

Note that a more complicated version also takes the center of mass into account [34], [35]. For simplicity, the pure rolling condition (i.e., no influence due to friction) is assumed. These equations are the basic ODEs for robot kinematics. As mentioned before, these ODEs can easily be converted to executable CBD models, allowing the robots to gain executable behaviour. This is shown in Fig 9, for the above equations. When a controller is added, the robots can move freely over the trajectories defined by this controller (which can be either "follow a line on the ground", or "follow this path defined by the warehouse", or "follow these actions by a remote control"...).

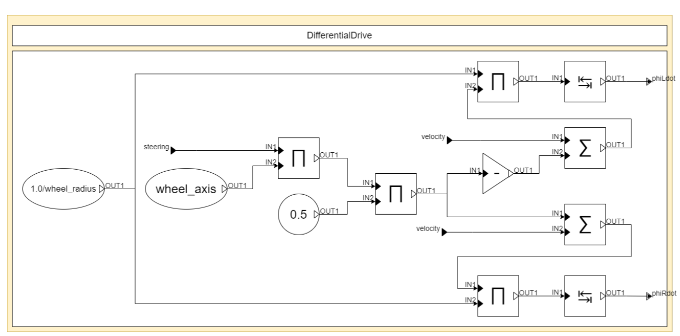

Fig. 9. CBD model of the AGV kinematic behaviour.

b) Timed Finite State Automaton (TFSA)

Besides the normal set of equations for the robot's movement, we should consider they are active in a warehouse. This implies that the robots should have *obstacle detection* (i.e., avoid driving into things) and *collision avoidance* (i.e., don't let multiple robots crash into each other). Collision avoidance can be predicted by the warehouse control system beforehand, whereas obstacle detection just happens. The controller algorithm needs to accommodate these situations as well.

Fig 10 shows a Timed Finite State Automaton (TFSA) that fully describes the robot behaviours. Inside the `Drive` state, a set of equations is embedded, as described in [36]. The basic functionality is simple: send the current location, check for obstacles, plan the next driving direction (i.e., $\dot{v}$ and $\dot{\omega}$) based on the current system state, and actually do the driving. As soon as some issue is encountered, the robot is halted until it receives a "restart" command.

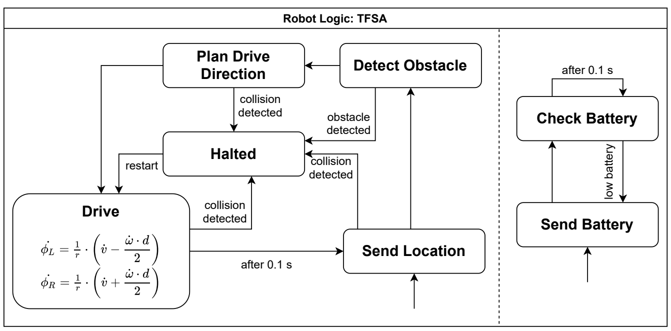

Fig. 10. TFSA model of the robot behaviour.

c) Robot Agent

Designing autonomous components requires modelling components as individual agents [28]. This is not always required in the system, as, in some cases, individual components need to get instructions from hierarchical entities that organize and control the system's workflow. However, autonomous entities can be designed at least to achieve predefined goals in the system and constantly work to fulfil them. For example, robots in a warehouse can be an example of autonomous entities that operate to fulfil the demands in the warehouse. The agent-based approach can be used to model the behaviour and characteristics of the robots [37]. The agent of a robot should incorporate all the models that form its internal and external behaviour (e.g., robot kinematics, navigation models, etc.). In other words, the robot agent is designed to control the robot by applying and using the provided models (from different aspects, behavioural and operational) to perform particular tasks.

The robot agent can operate in a sensing/acting manner or proactively according to goal-oriented behaviour. So, the robot's sensors are responsible for sensing the environment; then, based on this information/data, the agent of the robot will take a set of actions that could result in constructing its actuators to move or stop in case of collision. A simple overview of the robot agent interacting with the warehouse system is given in Fig. 11.

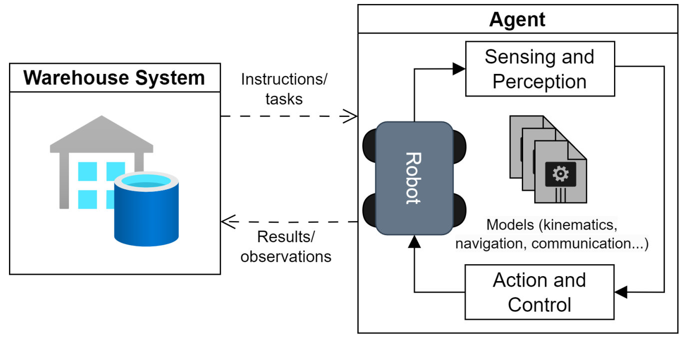

Fig. 11. Robot agent model.

In summary, several advantages are brought to the warehouse system and the robot by designing them based on an agent-based approach, such as: (1) achieving efficient cooperation and coordination between the system components, (2) making the system more flexible and adaptable to changes as agents can be active, reactive or proactive in different situations, (2) minimising the risks of a single-point failure and increase robustness as the system is partially decentralized, (4) increase autonomy and support local data processing as the robots can process a large amount of data locally and without overwhelming the control system of the warehouse, and also, (5) achieve scalability in the warehouse, so, the system can be scaled up easily by adding new robots or other components as agents.

2) Warehouse

The warehouse system should provide several tasks to enable the robots in the warehouse to navigate the environment safely, operate and collaborate with other system components efficiently, and accelerate to fulfil their tasks correctly and optimally. Any issues in operation can lead to delays and impact the whole system [38]. Hence, we provide some of the services the warehouse should provide to manage and administrate operations efficiently.

a) Task Allocation

Any warehouse will have multiple robots operating in the environment based on demand and overload. However, robots should be able to communicate with the warehouse system control to inform their availability (idle or busy), so the system control can decide for which closest and most suitable robot the task can be assigned. The sequence diagram for the task allocation is provided in Fig. 12.

b) Collision Avoidance

In many cases, predicting a possible collision between different moving robots requires installing several sensors in the robots and the environment to enable these robots to detect such a situation [39]. In our case, we used a minimalist number of sensors. The UWB, installed in the environment as anchors and as tags in the robots, are the main parts to detect such a scenario. Robots can drive and get updates from the UWB. Then, the control system aggregates this information and finds

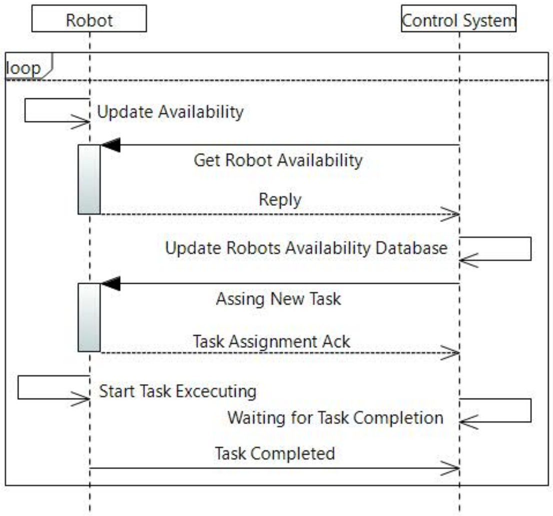

Fig. 12. Sequence diagram of the task allocation process in the warehouse.

which robots are close to each other. Fig. 13 depicts the sequence diagram of the collision avoidance example and how it is instrumented by the warehouse control system.

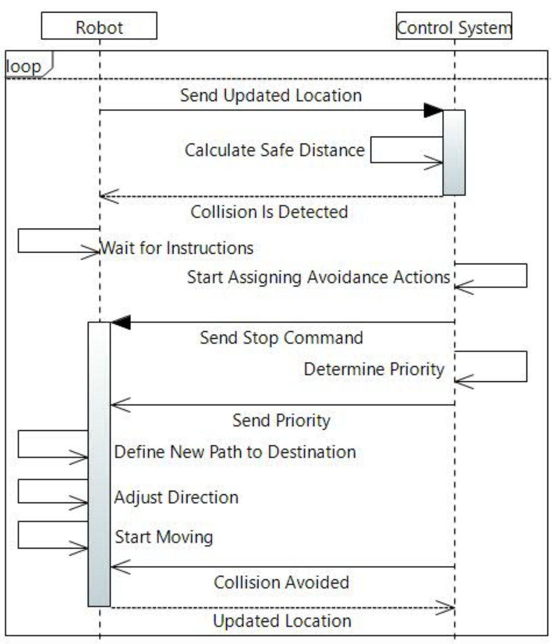

Fig. 13. Sequence diagram of the collision avoidance in the warehouse.

c) Power Management

As the energy resources in mobile systems such as robots are limited, thus, the need for efficient power and energy management is crucial to ensure better performance and reduce the significant costs of operating several robots. However, several algorithms, methods, and approaches exist in the literature to implement such mechanisms.

We provide one of the simple scenarios to implement such a method in a Multi-robot Warehouse system. In our example, three factors have been considered, the current battery level of the robot, the closest charging station for that robot, and, eventually, the availability of this charging station. The next Fig. 14 illustrates the approach followed in the warehouse system.

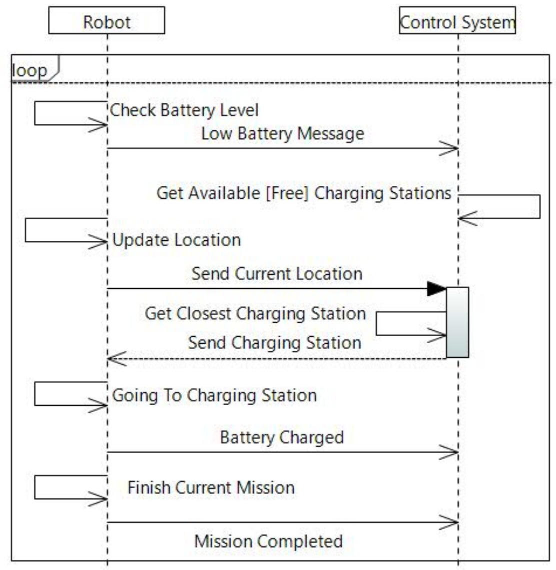

Fig. 14. Sequence diagram of the warehouse power management mechanism.

V. Conclusion & Future Work

In conclusion, an exemplary description of using MPM for a Multi-robot System use case is the focus of this paper. The design, implementation and deployment phases of the Multi-robot Warehouse system have considered different modelling formalisms that describe the structure, architecture and behaviour of the different components in the system. The Multi-robot System is complex and composed of heterogeneous and multidisciplinary components. Thus, modelling the different aspects of the system at their most appropriate level(s) of abstraction enables us to deal with the complexity of every level independently.

The models used in this exemplar paper are collected and classified according to the workflow activities described by the FTG+PM formalism to obtain the Multi-robot Warehouse.

Although this paper essentially serves as a reference for the collection of multi-models of the Multi-robot Warehouse system, the work conducted and the formalisms utilized to model and design the system could be extended, adapted and modified in future research projects that might consider replicating this work using other approaches, methods and formalisms.

REFERENCES

[1] Y. Z. Lun, A. D'Innocenzo, F. Smarra, I. Malavolta, and M. D. Di Benedetto, "State of the art of cyber-physical systems security: An automatic control perspective," *Journal of Systems and Software*, vol. 149, pp. 174–216, 2019.

[2] P. J. Mosterman and H. Vangheluwe, "Computer automated multi-paradigm modeling: An introduction," *Simulation*, vol. 80, pp. 433–450, Sept. 2004.

[3] 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.

[4] N. Pinkam, F. Bonnet, and N. Y. Chong, "Robot collaboration in warehouse," in *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, pp. 269–272, IEEE, 2016.

[5] Y. Wu and D. Ge, "Key technologies of warehousing robot for intelligent logistics," in *The First International Symposium on Management and Social Sciences (ISMSS 2019)*, pp. 79–82, Atlantis Press, 2019.

[6] T. Le-Anh and M. De Koster, "A review of design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 171, no. 1, pp. 1–23, 2006.

[7] M. Glatt, C. Sinnwell, L. Yi, S. Donohoe, B. Ravani, and J. C. Aurich, "Modeling and implementation of a digital twin of material flows based on physics simulation," *Journal of Manufacturing Systems*, vol. 58, pp. 231–245, 2021.

[8] J. Liu, J. Liu, C. Zhuang, Z. Liu, and T. Miao, "Construction method of shop-floor digital twin based on mbse," *Journal of Manufacturing Systems*, vol. 60, pp. 93–118, 2021.

[9] H. Zhang, Q. Qi, and F. Tao, "A multi-scale modeling method for digital twin shop-floor," *Journal of Manufacturing Systems*, vol. 62, pp. 417–428, 2022.

[10] X. Yang, X. Liu, H. Zhang, L. Fu, and Y. Yu, "Meta-model-based shop-floor digital twin architecture, modeling and application," *Robotics and Computer-Integrated Manufacturing*, vol. 84, p. 102595, 2023.

[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 Nature Singapore Singapore, 2023.

[12] H. Marah and M. Challenger, "Intelligent agents and multi agent systems for modeling smart digital twins," *Engineering multi-agent systems*, 2022.

[13] D. S. Losvik and A. Rutle, "A domain-specific language for the development of heterogeneous multi-robot systems," in *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 549–558, IEEE, 2019.

[14] M. Vještica, V. Dimitrieski, M. M. Pisarić, S. Kordić, S. Ristić, and I. Luković, "Production processes modelling within digital product manufacturing in the context of industry 4.0," *International Journal of Production Research*, pp. 1–20, 2022.

[15] O. B. Alhas, M. Oruc, G. B. Usta, H. Marah, B. T. Tezel, and M. Challenger, "Towards developing a cyber-physical warehouse system for automated order-picking for online shopping," in *Annals of computer science and information systems.-[Sl]*, vol. 32, pp. 321–328, 2022.

[16] E. Schoofs, J. Kisaakye, B. Karaduman, and M. Challenger, "Software agent-based multi-robot development: A case study," in *2021 10th Mediterranean conference on embedded computing (MECO)*, pp. 1–8, IEEE, 2021.

[17] M. E. Can, P. Ramkisoen, B. Karaduman, S. Demeyer, and M. Challenger, "Enhancing autonomous guided robots using software agents and uwb technology," in *11th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–6, IEEE, 2022.

[18] B. Karaduman, B. T. Tezel, and M. Challenger, "Deployment of software agents and application of fuzzy controller on the uwb localization based mobile robots," in *International Conference on Intelligent and Fuzzy Systems*, pp. 98–105, Springer International Publishing Cham, 2022.

[19] H. Ltaief, B. Karaduman, B. Boussaid, and M. Challenger, "Agent based implementation of a robot arm and smart production line using jade framework," in *11th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–12, IEEE, 2022.

[20] H. Vangheluwe, *Multi-formalism Modelling and Simulation*. PhD thesis, Ghent University, 2000.

[21] H. Vangheluwe, "DEVS as a Common Denominator for Multi-Formalism Hybrid Systems Modelling," in *IEEE International Symposium on Computer-Aided Control System Design*, (Anchorage, AK, USA), pp. 129–134, 2000.

[22] H. Nilsson, J. Peterson, and P. Hudak, "Functional hybrid modeling," in *International Symposium on Practical Aspects of Declarative Languages*, pp. 376–390, Springer, 2003.

[23] R. Paredis, J. Denil, and H. Vangheluwe, "Specifying and executing the combination of timed finite state automata and causal-block diagrams by mapping onto devs," in *2021 Winter Simulation Conference (WSC)*, 2021.

[24] C. Gomes, C. Thule, P. G. Larsen, J. Denil, and H. Vangheluwe, "Co-simulation of Continuous Systems: A Tutorial," Tech. Rep. arXiv:1809.08463 [cs, math], Sept. 2018.

[25] K. J. Åström, H. Elmqvist, and S. E. Mattsson, "Evolution of Continuous-Time Modeling and Simulation," in *Proceedings of the 12th European Simulation Multiconference, ESM'98*, pp. 9–18, ESM, 1998.

[26] C. Gomes, J. Denil, and H. Vangheluwe, *Causal-Block Diagrams: A Family of Languages for Causal Modelling of Cyber-Physical Systems*, ch. 4, pp. 97–125. Springer International Publishing, 2020.

[27] E. Fehlberg, "Klassische Runge-Kutta-Formeln fünfter und siebenter Ordnung mit Schrittweiten-Kontrolle," *Computing*, vol. 4, 1969.

[28] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *Ieee Access*, vol. 6, pp. 28573–28593, 2018.

[29] M. Challenger, K. Vanherpen, J. Denil, and H. Vangheluwe, "Ftg+pm: Describing engineering processes in multi-paradigm modelling," in *Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems*, pp. 259–271, Springer, Cham, 2020.

[30] R. Paredis, J. Exelmans, and H. Vangheluwe, "Multi-Paradigm Modelling for Model-Based Systems Engineering: Extending the FTG+PM," in *2022 Annual Modeling and Simulation Conference (ANNSIM)*, 2022.

[31] H. E. Plesser, "Reproducibility Vs. Replicability: A Brief History Of A Confused Terminology," *Frontiers in neuroinformatics*, vol. 11, p. 76, 2018.

[32] R. Paredis and H. Vangheluwe, "Exploring A Digital Shadow Design Workflow By Means Of A Line Following Robot Use-Case," in *2021 Annual Modeling and Simulation Conference (ANNSIM)*, 2021.

[33] K. M. Lynch and F. C. Park, "Wheeled Mobile Robots," in *Modern Robotics: Mechanics, Planning, and Control*, pp. 513–564, Cambridge University Press, 2017.

[34] E. O. Cobos Torres, S. Konduri, and P. R. Pagilla, "Study of wheel slip and traction forces in differential drive robots and slip avoidance control strategy," in *2014 American Control Conference*, pp. 3231–3236, 2014.

[35] S. Nandy, S. N. Shome, R. Somani, T. Tanmay, G. Chakraborty, and C. S. Kumar, "Detailed slip dynamics for nonholonomic mobile robotic system," in *2011 IEEE International Conference on Mechatronics and Automation*, pp. 519–524, 2011.

[36] R. Paredis, J. Denil, and H. Vangheluwe, "Specifying and executing the combination of timed finite state automata and causal-block diagrams by mapping onto devs," in *2021 Winter Simulation Conference (WSC)*, 2021.

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

[38] A. Dhaliwal, "The rise of automation and robotics in warehouse management," *Transforming Management Using Artificial Intelligence Techniques*, pp. 63–72, 2020.

[39] 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.