Decreasing the Design Complexity of Cyber-Physical Systems through Conceptual Modeling: The Robot Abstraction Method

Alexander Voelz
Faculty of Computer Science
Doctoral School Computer Science
University of Vienna
Vienna, Austria
Email: alexander.voelz@univie.ac.at

Daniel Kiesenhofer
Faculty of Computer Science
Research Group Knowledge Engineering
University of Vienna
Vienna, Austria
Email: a01251851@unet.univie.ac.at

Dimitris Karagiannis
Faculty of Computer Science
Research Group Knowledge Engineering
University of Vienna
Vienna, Austria
Email: dk@univie.ac.at

***Abstract*—The increasing complexity of designing, deploying, and maintaining Cyber-Physical Systems (CPS), particularly those incorporating multiple interacting robots, presents significant challenges regarding programming and system integration. Existing methods and tools that aim to decrease the complexity of setting up such systems through modeling-based approaches often focus on single-robot interaction and rely on manual data entries, thus limiting their scalability and applicability to more intricate multi-robot environments. This contribution introduces the Robot Abstraction Method (RobAM), a conceptual modeling method developed to decrease the design complexity of CPS through additional layers of abstraction. By utilizing conceptual modeling, capabilities of humanoid robots can be leveraged for the design of multi-robot interactions. Thereby, our approach enhances the modeling of complex CPS as well as the subsequent generation of code to enable efficient deployments. Through an experimental scenario, we demonstrate how RobAM simplifies the development process of CPS, paving the way for advancements regarding the integration of AI-supported technologies, such as Large Language Models, to ultimately generate contextually relevant semantics for future system design.**

I. Introduction

Historically, physical systems were designed to support humans with manual tasks or replace the need for human involvement altogether. The focus was on the hardware components and their respective functionalities. The corresponding software components of such systems were mainly used to control the physical capabilities rather than for communication and integration with other devices or whole systems. The technological advancements going along with the current era of digital transformation have caused an increasing interdependency between physical and digital components, partially due to the vast amounts of data that are being generated by and exchanged between devices within a system. These developments have formed the foundation for Cyber-Physical Systems (CPS), in which physical components interact with digital applications through communication channels.

Central to many CPS are robots equipped with diverse sensors and actuators that enable them to perform complex tasks autonomously and collaboratively. These systems inherently integrate Internet of Things (IoT) technologies, which provide the necessary connectivity and data exchange mechanisms. Modeling frameworks can provide relevant support for the development of these systems by facilitating intuitive programming and interaction paradigms. However, existing frameworks are primarily designed for single-robot systems and often require extensive domain knowledge, thus posing limitations when applied to complex CPS involving multiple interacting robots. As the complexity of these systems increases, the need for more intuitive and user-friendly modeling tools with code-generation capabilities also increases.

To address this need, we introduce the Robot Abstraction Method (RobAM), a modeling tool that aims to decrease the complexity of programming CPS with multiple humanoid robots. For this purpose, RobAM realizes additional layers of abstraction, enabling the modeling and deploying of complex CPS. Moreover, ongoing improvements of RobAM leverage Large Language Models (LLMs) to generate contextually relevant semantics based. The LLMs are utilized to interpret system requirements, interactions, and human feedback, allowing for more intuitive and adaptable modeling.

Based on these considerations, the contribution introduces relevant technologies, the fundamentals of modeling method design, and related work in Section II. Subsequently, our approach of utilizing abstraction to decrease the complexity of realizing the interaction between humanoid robots within CPS is presented in Section III. This includes detailed elaborations on future work, as the modeling method is currently Work in Progress (WIP). Finally, the key takeaways from this contribution are summarized in Section IV.

II. Theoretical Background

The theoretical background covers relevant concepts from the literature and the fundamentals regarding the intended realization of a modeling method. Afterward, existing solutions for programming such CPS in a visual and model-based manner are contrasted within the related work section.

A. Cyber-Physical Systems, Internet of Things, and Robots

The convergence of CPS and IoT represents a transformative paradigm in modern technology, enabling the seamless integration of computational and physical processes across diverse domains [1]. IoT encompass networks of physical objects embedded with sensors, actuators, software, and other technologies, enabling them to collect and exchange data [2]. These interconnected networks are often viewed as System of Systems (SoS), where various independent subsystems work together to achieve broader objectives, providing a scalable and interoperable framework [3]. The resulting connectivity is fundamental to CPS, as it allows for the continuous flow of information between physical and digital realms, facilitating real-time monitoring, control, and optimization. As an SoS, the integration of IoT with CPS creates complex systems capable of managing diverse and distributed components in real-time. Consequently, robots focused on supporting Human-Robot Interaction (HRI) [4] form one of many examples highlighting the integration of CPS and IoT. These robots are equipped with various IoT devices like sensors and actuators that enable them to interact with their environment. In this context, a domain-agnostic view of CPS has been proposed by Bagheri et al. [5], who define them as "*systems in which natural and human made systems (physical space) are tightly integrated with computation, communication and control systems (cyber space).*" For the remainder of this contribution, we thus assume the following CPS instantiation:

- Humans are natural entities of the physical space.
- Humanoid robots are human-made systems of the physical space.
- IoT devices are control systems enabling the integration between physical space and computation.
- Natural language and wireless data transfer are the communication methods used between systems.

B. Realization of Domain-Specific Modeling Methods

In the context of this contribution, two fundamentals from the conceptual modeling domain are relevant for the realization of the modeling method RobAM: the Generic Modeling Method Framework (GMMF) [6] and Agile Modeling Method Engineering (AMME) [7]. The first defines a blueprint for designing modeling methods, while the latter provides a guiding lifecycle for their development and iterative refinement.

Within conceptual modeling, abstraction is used to simplify a system under study, often as diagrammatic models [8]. The creation of such abstracted representations is enabled by a *modeling language*, which forms one of three integral components of a modeling method according to the GMMF. The modeling language defines the syntax, notation, and semantics of created models, with a corresponding metamodel capturing the conceptual architecture of the language as machine-processable structure [6]. The second component of the GMMF is the *modeling procedure* that stipulates how to apply the respective language. Lastly, *mechanisms and algorithms* provide functionalities that utilize components of the modeling language to enable extended capabilities. For the context of this contribution, past utilization of mechanisms and algorithms for code generation and operating cyber-physical components [9] provide guiding showcases of how Domain-Specific Modeling Methods (DSMMs) allow the modeling of and adapting to changing environments.

Considering CPS, domain-specific modeling can reduce the system's complexity and increase development productivity. Different approaches exist in the literature for modeling CPS or components of it, and corresponding methods are contrasted in the following subsection in terms of their applicability regarding robot-robot interaction as well as HRI. The development of a DSMM for realizing CPS in a model-based manner requires a metamodeling development and configuration platform. ADOxx[1] is such a metamodeling platform that supports the development of DSMMs concerning the above-mentioned components of the GMMF. Moreover, AMME provides a structured lifecycle to realize modeling methods. This lifecycle, comprising five phases—Create, Design, Formalize, Develop, and Deploy—guides the realization and iterative improvement of DSMMs [7].

C. Related Work

The related work section elaborates on existing languages, methods, and tools for realizing humanoid robots within CPS applications while emphasizing their respective limitations.

1) Choregraphe: Choregraphe[2] is a domain-specific modeling tool developed by SoftBank Robotics for programming the behavior of their humanoid robots, such as NAO, Pepper, and Whiz. The tool allows users to create models that generate code for these robots that are equipped with sensors and actuators to interact with the physical world. For example, the actuators of NAO cover arms, legs, joints, and the head, while the sensors include visual, touch, and acoustic components (cf., Section III-B). However, Choregraphe has several limitations, which are summarized in Table I.

2) RoboStudio: RoboStudio offers a visual programming environment focused on robots in the healthcare sector [10]. It enables healthcare personnel not familiar with programming to design and develop workflows for service robots that assist elderly people with daily routines. The environment leverages a Java-based technology stack to create and manage user interfaces and events that control the robot's actions. Corresponding limitations are documented in Table I.

3) PRINTEPS: PRINTEPS[3] stands for PRactical INTElligent aPplicationS, a framework developed by acquainted researchers at the Keio University for implementing robotic-based AI applications in a human-centric manner [11]. It is structured with a layered architecture that includes a service layer, process layer, and module layer. This design allows non-expert users to model workflows on a high abstraction level and detail them through successive layers, which are ultimately translated into executable code via libraries and tools

[1]Available at: https://www.adoxx.org/
[2]See also: http://doc.aldebaran.com/2-5/software/choregraphe/
[3]See also: https://printeps.org/

TABLE I
COMPARISON OF LIMITATIONS IN EXISTING MODEL-BASED APPROACHES FOR PROGRAMMING HUMANOID ROBOTS

Limitation	Choregraphe	RoboStudio	PRINTEPS
Interaction Modeling	Modeling interactions for a single robot, not supporting multi-entity interactions.	Modeling human-robot interactions, not supporting robot-robot interactions.	Modeling multi-layered human-robot interactions, not supporting direct robot-robot interactions.
Level of Abstraction	Limited to specific SoftBank robots, inadequate for various robots in a CPS.	Limited to screen-based interactions, inadequate for different modalities.	Lacks system-wide interaction and communication among multiple robots.
Adaptability	Proprietary, not open-source, limiting adaptability for diverse applications.	Proprietary, not open-source, limiting adaptability for diverse applications.	Needs additional layers to manage deployment across multiple entities.
Deployment	Designed for single robots, lacking multi-robot deployment mechanisms.	Designed for single robots, lacking multi-robot deployment mechanisms.	Designed for single robots, requiring separate models for the deployment of each additional robotic system.
Integration	External API integration available, not offering LLM-based generation features.	External API integration available, not offering LLM-based generation features.	External API integration available, not offering LLM-based generation features.

offered by the Robot Operating System (ROS) framework [12]. Nevertheless, similar limitations to those of Choregrpahe and RoboStudio remain (see Table I).

III. UTILIZING CONCEPTUAL MODELING TO DECREASE THE DESIGN COMPLEXITY OF PROGRAMMING HUMANOID ROBOTS WITHIN CYBER-PHYSICAL SYSTEMS

Considering the problem statement formulated within the introduction and the insights from the theoretical background (cf., Section I, II), this research is framed as a design science artifact [13]. Its goal is to develop a DSMM for decreasing the complexity of designing and programming CPS that involve multiple interacting robots. The development of the DSMM follows the Design Science Research (DSR) process model proposed by Peffers et al. [14] while adhering to the GMMF and AMME lifecycle introduced in Section II-B. Limitations of existing approaches (cf., Table I) form the foundation for defining objectives of a corresponding solution to the formulated problem. The following subsections are dedicated to the remaining stages of the DSR methodology, each addressing a critical phase in the realization and refinement of our proposed solution. The specific requirements for achieving the objective of an intuitive modeling tool with code-generation capabilities are subsequently presented (see also Table II).

TABLE II
REQUIREMENTS OF A DSMM FOR PROGRAMMING HUMANOID ROBOT INTERACTIONS WITHIN CPS DERIVED FROM EXISTING APPROACHES

Limitation	Derived DSMM Requirement
Interaction Modeling	• Modeling of robot-robot interactions & HRI • Support for multiple cyber-physical entities
Level of Abstraction	• High-level system abstraction • Flexible device/communication modeling
Adaptability	• Extensible modeling method • Open-source framework
Deployment	• Multi-entity deployment capabilities • Model-based code-generation capabilities
Integration	• API integration • LLM-based generation capabilities

Regarding *Interaction Modeling*, the fundamental goal of RobAM is to enable the modeling of robot-robot interaction and also HRI. Consequently, it must be possible to model the behavior of multiple interacting entities and the method of communication they use for the respective interactions.

Although this contribution focuses on humanoid robots as interacting entities within a CPS, the modeling method shall still provide a *Level of Abstraction* so that it can be adapted to other types of human-made systems of the physical space (i.e., other types of robots with different capabilities). The same applies to the potential adaption of communication methods. While the paper covers the interaction of humans with robots and robots with each other via natural language and Wi-Fi, various other types of communication (e.g., wired or web-based) exist that need to be considered in future works.

Adaptability is an imperative notion in complex and changing environments and has to be ensured at all levels of the modeling method. For instance, modeling elements, such as cyber-physical entities and their capabilities, must be extensible to allow for a wider range of entities and communication channels. This kind of adaptability is facilitated through every iteration of the AMME lifecycle. By utilizing the ADOxx metamodeling platform in the context of the OMiLAB Community of Practice [15], the modeling method will ultimately be available in an open-source format.

The RobAM method needs to support multi-robot and also multi-entity *Deployment* mechanisms. Furthermore, the created models shall directly serve as corresponding implementations for the modeled system, thus requiring the integration of code generation capabilities into the modeling method. This in turn allows users without a background in programming to design and implement CPS containing interacting robots.

To harness new technologies and innovations, *Integration* forms a crucial requirement for any information system. The existing approaches that were compared all offer some form of external API integration, while LLM-based generation features are not directly available. RobAM has the aim to integrate LLM-based functionalities for context-aware model generation, while also offering API integration to encourage adaptions and extensions following the open-source principle.

A. Robot Abstraction Method (RobAM)

This section details the *Design and Development* stage of our proposed modeling method, RobAM, which aims at addressing the previously identified requirements (cf., Table II). Subsequently, the fundamentals of this method are explained, covering the implemented layers of abstraction.

As a first layer of abstraction, RobAM utilizes the ROS library, which offers open-source implementations for a multitude of robotic systems. In the same way, PRINTEPS (cf., Section II-C3) builds a modeling framework on top of ROS, allowing to capture robot-based capabilities as models that enable code generation. For this reason, the PRINTEPS method is utilized as intermediate output to reuse established code generation capabilities through model transformation mechanisms, thus satisfying the second *Deployment* requirement. Still, it has been mentioned that PRINTEPS is limited to modeling one robotic system at a time (cf., Table I). To address this limitation, RobAM builds upon the existing implementation by further abstracting from it to support the modeling of interactions between robotic systems. The resulting layers of abstraction are divided into:

- The *Presentation Layer* provides an overview of the complete CPS and all its components. Within this overview, relations between represented CPS entities and their respective communication channels are defined.
- Models from the *Interaction Layer* are referenced in the *Presentation Layer* and specify the choreography of CPS members. Choreography describes the decentralized coordination of services [16], which in this case are interactions between multiple CPS entities. The *Interaction Layer* thus forms the foundation for concrete workflows that are executed within a given scenario.
- The modeling of CPS-based interactions usually requires domain-specific knowledge, which is captured in the separated *Content Layer*. Corresponding content models thereby represents the knowledge base of a given CPS, which are referenced in the *Presentation Layer* to be accessed within the choreography of CPS entities.

A conceptual overview of RobAM covering these three layers is displayed in Fig. 1. Exemplary models of each layer are displayed separately in Fig. 2, which are based on the folowing teacher and student case. On the very left side of Fig. 1, ROS offers the lowest level of abstraction by directly programming humanoid robots based on the commands they can execute (e.g., speak, listen walk, grab). PRINTEPS enables the modeling of such capabilities while providing the aforementioned code generation capabilities. RobAM further abstracts from these capabilities by offering the presented abstraction layers that support the modeling of robot-robot interactions and the subsequent model-based code generation. The LLM icon next to the *Content Layer* and *Interaction Layer* indicates which models are planned to be generated on the basis of generative Artificial Intelligence technologies like LLMs, as outlined in Section III-C. The next section details an example of applying RobAM in a specific scenario.

B. Teacher and Student Case based on RobAM

For the *Demonstration* stage of our method, a teacher and student case is utilized, serving as a practical example that illustrates how RobAM can be applied to real-world scenarios. Before going into further detail, the setup of the scenario has to be clarified. For the teacher and student case, two NAO robots are employed. To interact with the physical world, robots like NAO use sensors to convert physical events into electrical signals and actuators to transform these signals into physical actions [17]. This is crucial for entities in a CPS as it enables interactions between the physical and cyber spaces. NAO robots utilize sensors to assess three types of events from the physical environment and several actuators located in three areas to perform physical actions[4]:

- Visual events: Equipped with visual sensors, NAO's eyes can recognize and localize objects in their surroundings. This capability is essential for identifying nearby interaction partners, such as humans or other robots.
- Haptic events: Touch sensors on NAO's head and arms detect physical contact. This is vital for HRI and tasks that involve handling objects.
- Acoustic events: NAO's integrated speech recognition module allows it to process natural language, enabling it to respond to verbal commands during interactions.
- Head: An actuator in the neck controls head movements, and the eyes can change color to provide visual signals, mimicking human-like behaviors such as nodding.
- Arms: Actuators in the arms and joints allow NAO to grab or carry objects and make gestures.
- Legs: Leg actuators enable NAO to walk to specific locations, like moving closer to a human for a conversation.

The teacher and student case based on this setup revolves around a scenario in which the NAO acting as a teacher asks the NAO acting as a student predefined questions and expects an answer to be given. To realize this scenario using RobAM, the overview of the CPS environment has to be modeled within the *Presentation Layer*. The corresponding *Presentation* model is displayed in Fig. 2a, in which the individual NAOs are represented as CPS members, each consisting of several components that specify their abstracted capabilities. Moreover, the communication method of the robots is specified. In our example, robot modules are used to recognize and respond to speech, while web modules enable the processing and analyzing of digitalized speech.

The specification of domain knowledge is captured within the *Content Layer*. The corresponding *Content* model of the teacher and student scenario consists of a root data module named "School Questionnaire", that contains the two submodules "Geography" and "Physics" (cf., Fig. 2b). Each of these topics references different data objects that contain specified questions with expected answers. In this way, questions can be accessed by the CPS member "NAO Teacher", as specified in the associated *Interaction* model.

[4]For more detailed information, see http://doc.aldebaran.com/2-1/family/nao_dcm/actuator_sensor_names.html

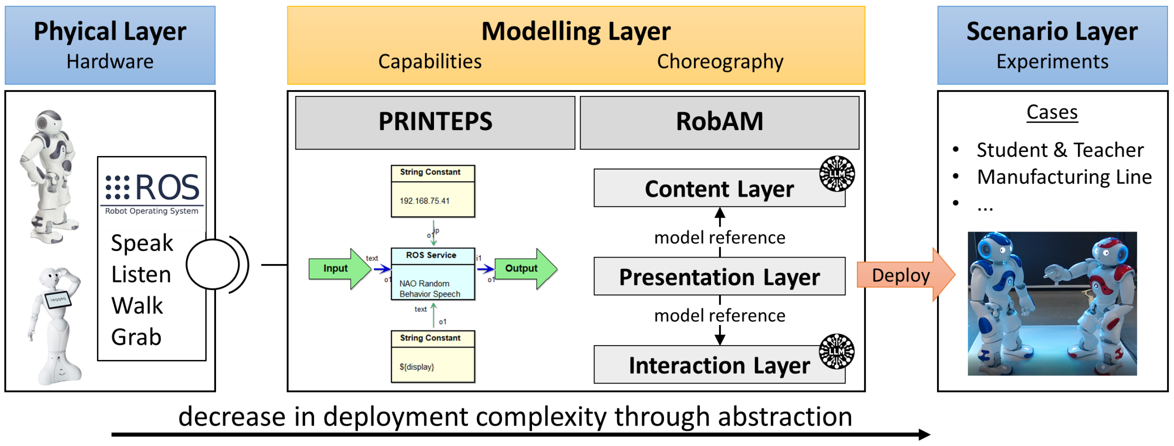


Fig. 1. Conceptual overview of the layered RobAM approach that builds upon ROS and PRINTEPS to decrease CPS design and deployment complexity (the LLM icon next to the *Content Layer* and *Interaction Layer* indicates ongoing work regarding the development of LLM-based model generation capabilities).

The *Interaction Layer* details the interaction choreography for each CPS member according to the modeled scenario. Fig. 2c and Fig. 2d showcase the *Interaction* models of the simplified choreographies for the CPS members “NAO Teacher” and “NAO Student”. The teacher choreography starts with the action “Ask First Question”, which references a question contained in the *Content* model to be expressed in natural language. A resulting request is received by the student waiting for instructions, which then triggers the answering of the question. Next, a request containing the given answer is received by the “NAO Teacher” followed by the action “Analyze Answer”. This sequence of events can be continued repeatedly until an end event is specified. In Fig. 2c and Fig. 2d, two repetitions of this sequence are modeled, with the request containing the student answer to the second question ending the choreography of the “NAO Student” while the subsequent analysis of this answer ends the choreography of the “NAO teacher”. It is important to note that actions like “Answer Question” and “Analysis of Answer” are adaptable according to the modeled interaction (e.g., human providing the answer; analysis of answer influencing the selection of the following question). An example of utilizing this adaptability is presented in the next section.

After all three layers of RobAM have been modeled, their content is merged through a specified model transformation mechanism that generates PRINTEPS models as intermediate output. This enables the subsequent generation of ROS code implementing the modeled scenario (cf., Section III-A).

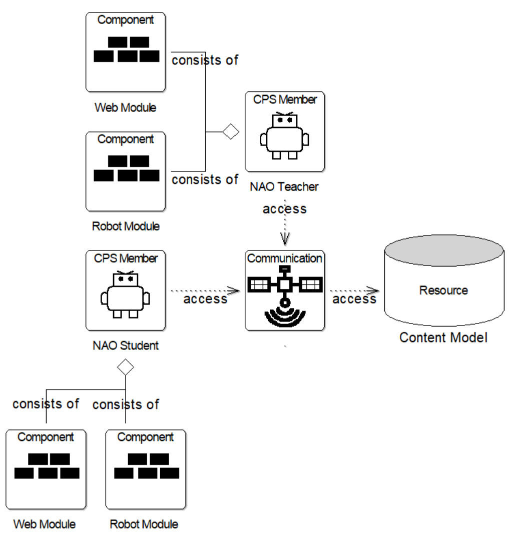
C. Future Work

The following section outlines the *Evaluation* stage, which serves as an assessment of RobAM. As part of the DSR methodology, it was decided that this stage iterates back to the *Design* stage (cf., [14]), allowing for the improvement of identified shortcomings through the integration of innovative technologies. Consequently, RobAM constitutes WIP still under development, and future work is discussed below.

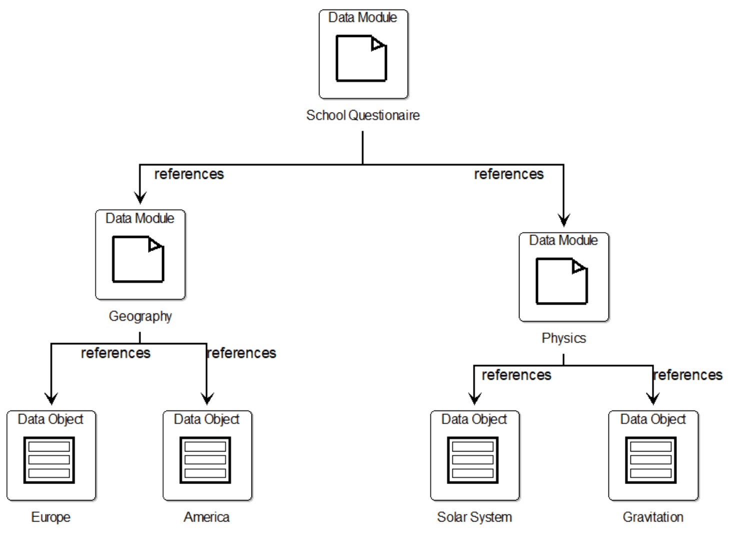
First, an automated generation of content models containing relevant domain semantics is being implemented through the utilization of LLMs combined with structured prompt engineering. These LLMs are built upon the Transformer model architecture [18], which provides the basis for their advanced capabilities of generating contextually relevant content. The goal of this extension is to reduce the complexity of programming the interaction between multiple CPS entities even further by eliminating the tedious task of creating *Content* models, especially regarding their population with relevant data. Currently, each question in the *Content* model that can be accessed by CPS members is manually added by a modeler.

Second, a similar approach of combing LLMs and prompt engineering is being utilized to automate the generation of *Interaction* models. The pseudocode shown in Algorithm 1 represents the corresponding realization of this functionality, considering that the specification of the required syntax will be provided within the prompt in the future. The interaction loop begins by generating an initial question and expected answer for a given topic, such as addition. After posing this question to the student and receiving an answer, the system validates the correctness of the response. If the answer is correct, a more advanced follow-up question is generated to further challenge the student. If the answer is incorrect, the LLM generates a follow-up question that considers the student’s mistake, providing a tailored learning experience that addresses specific misunderstandings. This iterative process continues until no more answer is provided.

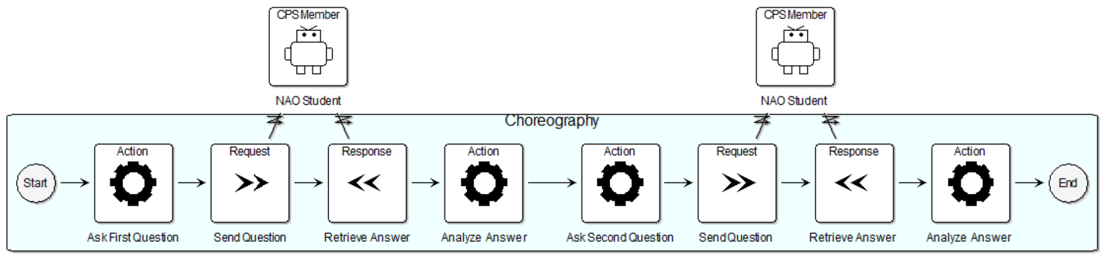
Third, the CPS member concept was extended as part of the previously elaborated refinements to capture relevant characteristics of the generated *Content* model under consideration of the corresponding *Presentation* and *Interaction* model. This has important implications for the application of RobAM in changing environments that require the utilization of varying robot capabilities. Considering the capabilities of NAO (cf., Section III-B), the following extensions of the teacher and student scenario are currently being investigated:



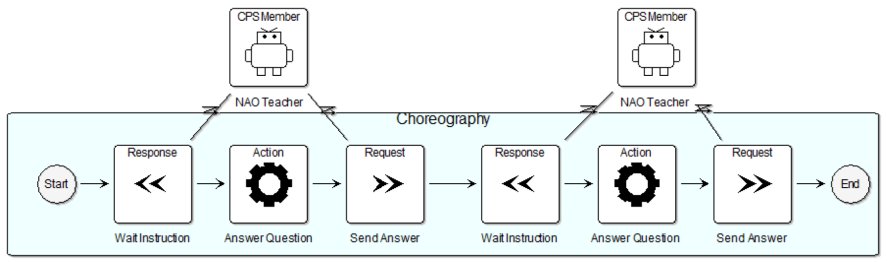
(a) Presentation model of the teacher and student case



(b) Content model of the teacher and student case



(c) Interaction model of CPS member “NAO Teacher”



(d) Interaction model of CPS member “NAO Student”

Fig. 2. Different model types within the realization of the teacher and student case based on the layered RobAM approach.

- The visual sensors can be used to recognize human emotions during the interaction with students. In line with affective computing theories [19], such recognized emotions can serve as valuable input for prompts that are utilized to generate follow-up questions.
- The combination of visual sensors and the multitude of actuators can enable interaction with several students. For example, a question could be posed to students who have to raise their hands to answer. The NAO teacher could detect these students, pick randomly among them, and walk to the respective students to assess their answers through its speech recognition modules.

Finally, the Communication stage of the DSR methodology is also part of future work. More precisely, when the last phase of the AMME lifecycle, *Deploy*, is completed, the results and insights gained from the research will be shared through the OMiLAB Community of Practice [15] in an open-source manner. This ensures that the developed method is accessible to a wider audience, allowing for further refinement and application of RobAM across various domains. Both the DSR methodology and the AMME lifecycle emphasize an iterative approach for guiding constant improvements and future developments of the modeling method, which constitutes WIP still under development. This includes exploring the integration of RobAM with the Digital Twin paradigm, enabling real-time mirroring of CPS environments for enhanced simulation and feedback. Furthermore, efforts will focus on ensuring the scalability of RobAM across various CPS scenarios, enabling it to manage increasingly complex systems with multiple interacting components. This iterative cycle will ensure that RobAM remains adaptable in evolving CPS environments.

Algorithm 1 Pseudocode for adaptive question generation in teacher and student interactions (Python-inspired)

```
Require: import LLM
  prompt = ”Create a question to exercise ” + topic
  question = LLM.generate(prompt)
  ask_student(question)
  student_answer = get_student_answer()
  while student_answer do
    prompt = ”Is ” + student_answer + ” the correct answer
    to the question ” + question + ”?”
    correct = LLM.generate(prompt)
    if correct then
      prompt = ”Create a follow-up question more advanced
      than ” + question + ” to practice ” + topic
    else
      prompt = ”Create an appropriate follow-up question
      to practice ” + topic + ” in a way that considers the
      student’s incorrect answer” + student_answer
    end if
    question = LLM.generate(prompt)
    ask_student(question)
    student_answer = get_student_answer()
  end while
```

IV. Conclusion

The complexity of realizing CPS with multiple interacting robots presents significant challenges. These challenges result from the extensive coding effort on a low abstraction level and the required adaptability to changing environments. Model-based approaches that address these challenges, such as Choregraphe, RoboStudio, and PRINTEPS, still have limitations.This contribution introduced RobAM, developed using the DSR methodology, along with the GMMF and AMME lifecycle. RobAM addresses the identified limitations by facilitating explicit multi-robot interactions, providing high-level system abstractions, supporting open-source adaptability, and enabling distributed deployments. For better understanding, an illustrative example of a teacher and student case was presented to showcase the application of RobAM.

The future work section presented current efforts of enhancing RobAM through the integration of LLM-based functionalities. The resulting approaches focus on the automated generation of (i) content models, and (ii) interaction models that consider student answers to generate suitable follow-up questions. By integrating such functionalities, future implementations can provide adaptive educational interactions, tailored to the individual student’s understanding. These advancements leverage the power of LLMs to dynamically generate contextually relevant content. In the broader context, these LLM-driven functionalities open up possibilities for RobAM to be applied in a variety of educational and CPS scenarios, following the ultimate goal of decreasing the design complexity of CPS through conceptual modeling.

References

[1] V. Lesch, M. Züfle, A. Bauer, L. Iffländer, C. Krupitzer, and S. Kounev, “A literature review of iot and cps—what they are, and what they are not,” *Journal of Systems and Software*, vol. 200, p. 111631, 2023.

[2] S. Li, L. D. Xu, and S. Zhao, “The internet of things: a survey,” *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[3] G. Fortino, C. Savaglio, G. Spezzano, and M. Zhou, “Internet of things as system of systems: A review of methodologies, frameworks, platforms, and tools,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 223–236, 2021.

[4] J. Guggemos, S. Seufert, and S. Sonderegger, “Humanoid robots in higher education: Evaluating the acceptance of pepper in the context of an academic writing course using the utaut,” *British Journal of Educational Technology*, vol. 51, no. 5, pp. 1864–1883, 2020.

[5] B. Bagheri, S. Yang, H.-A. Kao, and J. Lee, “Cyber-physical systems architecture for self-aware machines in industry 4.0 environment,” *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 1622–1627, 2015.

[6] D. Karagiannis and H. Kühn, “Metamodelling platforms,” in *E-Commerce and Web Technologies*, K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, Eds. Berlin Heidelberg: Springer, 2002, p. 182.

[7] D. Karagiannis, “Agile modeling method engineering,” in *Proceedings of the 19th Panhellenic Conference on Informatics*, N. Karanikolas, D. Akoumianakis, M. Nikolaidou, D. Vergados, and M. Xeno, Eds. New York, NY, USA: Association for Computing Machinery, 2015, p. 5–10.

[8] H. C. Mayr and B. Thalheim, “The triptych of conceptual modeling,” *Software and Systems Modeling*, vol. 20, pp. 7–24, 2021.

[9] D. Karagiannis, R. A. Buchmann, P. Burzynski, U. Reimer, and M. Walch, *Fundamental Conceptual Modeling Languages in OMiLAB*. Cham: Springer International Publishing, 2016, pp. 3–30.

[10] C. Datta, C. Jayawardena, I. H. Kuo, and B. A. MacDonald, “Robostudio: A visual programming environment for rapid authoring and customization of complex services on a personal service robot,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2352–2357.

[11] T. Morita, S. Akashiba, C. Nishimoto, N. Takahashi, R. Kukihara, M. Kuwayama, and T. Yamaguchi, “A practical teacher—robot collaboration lesson application based on printeps,” *The Review of Socionetwork Strategies*, vol. 12, pp. 97–126, 2018.

[12] T. Morita, K. Nakamura, H. Komatsushiro, and T. Yamaguchi, “Printeps: An integrated intelligent application development platform based on stream reasoning and ros,” *The Review of Socionetwork Strategies*, vol. 12, pp. 71–96, 2018.

[13] R. J. Wieringa, *Design science methodology for information systems and software engineering*. Springer, 2014.

[14] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, “A design science research methodology for information systems research,” *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007.

[15] A. Völz and I. Vaidian, “Digital transformation through conceptual modeling: The nemo summer school use case,” in *Modellierung 2024*. Bonn: Gesellschaft für Informatik e.V., 2024, pp. 139–156.

[16] T. Cerny, M. J. Donahoo, and M. Trnka, “Contextual understanding of microservice architecture: current and future directions,” *SIGAPP Applied Computing Review*, vol. 17, no. 4, p. 29–45, 2018.

[17] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Actuators and Sensors*. London: Springer London, 2009, pp. 191–231.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017, pp. 5998–6008.

[19] C.-H. Wu, Y.-M. Huang, and J.-P. Hwang, “Review of affective computing in education/learning: Trends and challenges,” *British Journal of Educational Technology*, vol. 47, no. 6, pp. 1304–1323, 2016.