Real-world verification techniques for Robotic and Embedded BDI agents: A systematic mapping

Bruno Policarpo Toledo Freitas[1,2], **Carlos Eduardo Pantoja**[1], **José Viterbo**[2]

[1]Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ)

[2]Universidade Federal Fluminense (UFF) - Instituto de Computação

{bruno.freitas,carlos.pantoja}@cefet-rj.br, viterbo@ic.uff.br

Abstract. *An agent is an autonomous software entity capable of cognition, proactivity, and adaptability. A group of agents collaborating, cooperating, or competing between themselves is called a Multiagent System (MAS). MAS are widely adopted for robotic and embedded systems as their natural autonomy is well suited to these systems. Given the costs associated with real hardware construction, it is a common practice to use simulators in their development before deploying to the real world. However, simulations may not get a full account of what will happen in the real world. This work presents an ongoing work on methods to verify and guarantee behaviors observed in simulations happen as well on the real world of embedded and robotic MAS.*

1. Introduction

An agent is an autonomous software entity capable of independence, cognition, and proactivity. A group of agents that interact among themselves is called a Multiagent System (MAS). A MAS is a common way to develop solutions to complex tasks as the agents can also show cooperation, collaboration or competition [Bordini and Hübner 2006, Bratman et al. 1988, Sichman et al. 1992]. Given is natural adaptability, MAS also have been used in robotics and embedded systems for a vast array of applications. For instance, it has been proposed solutions to control an automotive assembly line [Montoya-Zapata et al. 2024], monitor industrial or urban zones [Vallejo et al. 2020].

A challenge that occurs in the development of any embedded or robotic system is the deployment of the system itself. Given the costs of real hardware construction, it is common to use simulators to avoid possible damages when developing the system on real hardware. However, while simulators are a good way to test and validate design ideas cheaply, their execution in the real world may encounter unexpected situations, such as hardware failures, which can, in turn, may deviate their execution from the expected. In this regard, the system designer is faced with two choices: he either trusts the system to run without supervision or takes measures to guarantee the behaviour observed happens as well on the deployed system.

This work presents a systematic mapping about methods to guarantee that behavior observed in MAS simulations occurs as well in the real world, with a focus on robotic and embedded systems. The rest of this paper is organized as follows. Section 2 presents the methodology adopted for this systematic mapping. Sectionr 3 presents a preliminary discussion of the works found on Section 2. Section 4 presents the conclusion.

2. Methodology

This section presents the specification of the systematic mapping proposed.

- **RQ1:** Which methods or technologies are used to guarantee in the real-world properties of embedded or robotic systems BDI MAS observed on simulators?
- **RQ2:** Do embedded or robotic BDI MAS usually have their behavior in simulations verified against real world execution?
- **RQ3:** Which MAS problem domains usually use simulators?
- **RQ4:** What are the main MAS simulators on each domain?

RQ1 and RQ2 are meant to discover current methods, formal or not, used to compare behaviors in the simulations with real-world execution. Additionally, while not meant as the primary goal, it is also intended to discover current methods to correct unexpected behaviors, as it can be a possible research direction. The RQ3 and RQ4 are meant to discover simulators commonly used in MAS development, as they can provide targets to implement and validate any method created.

Given these questions, we proceed to search the articles databases, using the following search string: *("Robotic"OR "Embedded"OR "cyber-physical") AND ("real world"OR "real-world") AND ("Belief Desire Intention"OR "BDI"OR "Belief-Desire-Intention")*

It has been selected works from the last 5 years from the bases in Table 2, which also shows the number of works found before and after applying the Inclusion and Exclusion criteria from Table 1 by reading abstracts. The search was conducted between September to November of 2025. It is intended, for the future, to run afterwads a snowballing based on these works.

Inclusion criteria	**Exclusion criteria**
The work proposes an Embedded or robotic MAS directly on the real world	Short paper
The work proposes an Embedded or robotic MAS using simulators	Books or chapter books
The work proposes an Embedded or robotic MAS, using simulators and then implementing on the real world	Works that don't employ MAS solutions
	Works without access by institution

Tabela 1. Inclusion and Exclusion criteria used

After these last steps, we continued filtering in the works by reading the Introduction followed by its Conclusion and applying the Quality Assessment below, in which were selected works with a minimum score of 1.5 points:

- **(1 point)** The works uses any method, formal or not, to guarantee properties observed in simulations when it is deployed on the real world?
- **(1 point)** The work compares results from simulations with the real world?
- **(1 point)** The work uses simulations?

Database	Number of works found	Number of works after applying inclusion/exclusion criteria
ACM Digital Library	98	16
IEEE Digital Library	3	1
Science@Direct	86	22
Scopus	23	10

Tabela 2. Summary of number of works found

3. Preliminary results

This section presents a brief description so far of the works filtered in by the criteria from Table 1. After applying them, 13 works have been selected, with 11 other works being blocked from access by the institution.

Digital Twins (DT) are a common term encountered for works that implement something in the real world. It has been proposed a way to improve the scheduling of a physical storage yard by creating a DT of it [Gao et al. 2022], in which the optimizations are run. Another interesting application connecting both the real and virtual world is presented by [Marah and Challenger 2024], which aims to represent DT of Multirobot Systems (MRS). It also implements a simulation platform representing the physical system and its DT, allowing the robot's control by its DT.

Some theoretical works have used as examples scenarios taken from robotic or embedded domains. For example, a work proposing unifying action and planning into a single model [Patra et al. 2021] used simulations to illustrate their ideas by identifying 5 types of scenario domains: Search & Rescue (S&R), Explore, Fetch, Navigate, and Deliver. Its descriptions follow below, along with its main characteristics in Figure 1. These scenarios can make interesting test cases to be used in future experimentations.

- S&R: a type of mission where robots must find and rescue injured people.
- Explore: a mission to discover and monitor a specific area.
- Fetch: the mission is collecting an Object of Interest.
- Navigate: a team of robots needs to move objects from a room to another
- Deliver: the team of robots must package incoming orders and deliver to customers

Features of the test domains.

Domain	Dynamic events	Dead ends	Sensing	Robot collaboration	Concurrent tasks
S&R	✓	✓	✓	✓	✓
Explore	✓	✓	✓	✓	✓
Fetch	✓	✓	✓	–	✓
Nav	✓	–	✓	✓	✓
Deliver	✓	✓	–	✓	✓

Figura 1. Features of Test domains [Patra et al. 2021]

Natural or man-made disasters need information retrieval and intelligent analyses, which in turn need autonomous agents to cooperate with each other. For this, a UAV MAS has been proposed [Vallejo et al. 2020] to make such monitoring, using as a

real-world example the monitoring of an industrial complex and an urban environment. Another application of the MAS solution to a real-world problem is to tackle disturbances in the kitting process of an automotive assembly line [Montoya-Zapata et al. 2024]. It also compares the results of its approach with data collected from a real automotive plant.

The uncertainty of the real world and its effects in Cyber-Physical Systems (CPS) is a widely known problem. A summary of possible problems that can arise during its development has been proposed [Staroletov et al. 2019], seen on Figure 2. As the author suggests, a code ran in simulations does not guarantee it will work on real world, and bugs on the firmware or hardware parts may also result in incorrect behavior as expected. For the proposal of this research, the main areas of study will concentrate from simulation and testing to the implementation itself.

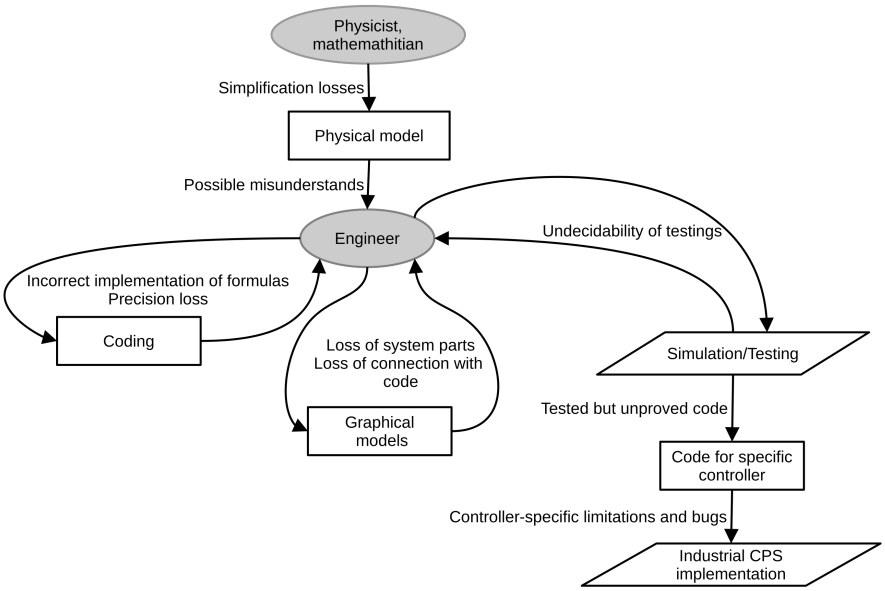

Figura 2. Possible problems during CPS development [Staroletov et al. 2019]

Fuzzy logic and symbolic AI techniques have been proposed to deal with uncertainties of CPS during executions presented in the last paragraph [Karaduman et al. 2024]. The work also applies to a real-world scenario, which is a smart production case study. Another solution, combining model checking and runtime verification, aims to avoid state space explosion and detect system assumptions violations [Ferrando et al. 2021], using as a motivating example a cruise control and a simulation of Mars Curiosity Rover with ROS [Robotics 2025] and Gazebo. The changes of physical resources during runtime have also been explored [Lazarin et al. 2024] by considering the scenarios of addition, fault tolerance, upgrade and swap, but without taking into account simulation results.

The problem of verification techniques for BDI agents grows in demand as such systems become increasingly complex. It has been proposed in the literature the use of BiGraphs [Archibald et al. 2022] to make such verification for BDI agents using the CAN language, using a UAV case study to demonstrate its use. A similar problem is testing with high coverage in complex systems, which is time-consuming and often too expensive to do. Co-verification has been proposed as a way to do pragmatic verification of system properties [Murray et al. 2022], in which models of software and hardware are coupled

through platform mappings that specify how the software and hardware are interconnected in terms of sensors and actuators. It uses a real-world example of a car painting machine, using RoboChart as the robot software simulator and Simulink as a hardware simulator.

4. Conclusion

Deploying robotic and embedded solutions to the real world is a challenging task that usually brings up issues that were not foreseen during development. In this regard, this work presented an ongoing systematic mapping on techniques to verify and guarantee that behaviours observed in simulations also happens in the real world.

Referências

Archibald, B., Calder, M., Sevegnani, M., and Xu, M. (2022). Modelling and verifying BDI agents with bigraphs. *Science of Computer Programming*, 215:102760.

Bordini, R. H. and Hübner, J. F. (2006). BDI Agent Programming in AgentSpeak Using Jason. In Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Toni, F., and Torroni, P., editors, *Computational Logic in Multi-Agent Systems*, volume 3900, pages 143–164. Springer Berlin Heidelberg, Berlin, Heidelberg. Series Title: Lecture Notes in Computer Science.

Bratman, M. E., Israel, D. J., and Pollack, M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(3):349–355.

Ferrando, A., Dennis, L. A., Cardoso, R. C., Fisher, M., Ancona, D., and Mascardi, V. (2021). Toward a Holistic Approach to Verification and Validation of Autonomous Cognitive Systems. *ACM Trans. Softw. Eng. Methodol.*, 30(4):43:1–43:43.

Gao, Y., Chang, D., Chen, C.-H., and Xu, Z. (2022). Design of digital twin applications in automated storage yard scheduling. *Advanced Engineering Informatics*, 51:101477.

Karaduman, B., Tezel, B., and Challenger, M. (2024). On the impact of fuzzy-logic based BDI agent model for cyber–physical systems[Formula presented]. *Expert Systems with Applications*, 238.

Lazarin, N. M., Pantoja, C. E., and Viterbo, J. (2024). Dealing with the Unpredictability of Physical Resources in Real-World Multi-agent Systems. In Rocha, A. P., Steels, L., and van den Herik, J., editors, *Agents and Artificial Intelligence*, pages 48–71, Cham. Springer Nature Switzerland.

Marah, H. and Challenger, M. (2024). Adaptive hybrid reasoning for agent-based digital twins of distributed multi-robot systems. *SIMULATION*, 100(9):931–957. Publisher: SAGE Publications Ltd STM.

Montoya-Zapata, S., Klement, N., Silva, C., Gibaru, O., and Lafou, M. (2024). Multi-agent system for perturbations in the kitting process of an automotive assembly line. *Engineering Applications of Artificial Intelligence*, 135:108679.

Murray, Y., Sirevåg, M., Ribeiro, P., Anisi, D. A., and Mossige, M. (2022). Safety assurance of an industrial robotic control system using hardware/software co-verification. *Science of Computer Programming*, 216:102766.

Patra, S., Mason, J., Ghallab, M., Nau, D., and Traverso, P. (2021). Deliberative acting, planning and learning with hierarchical operational models. *Artificial Intelligence*, 299:103523.

Robotics, O. (2025). ROS: Home.

Sichman, J. S., Demazeau, Y., and Boissier, O. (1992). When can knowledge-based systems be called agents? In *Proceedings of the IX Brazilian Symposium on Artificial Intelligence (SBIA)*, volume 9, pages 172–185, Rio de Janeiro. SBC.

Staroletov, S., Shilov, N., Zyubin, V., Liakh, T., Rozov, A., Konyukhov, I., Shilov, I., Baar, T., and Schulte, H. (2019). Model-driven methods to design of reliable multiagent cyber-physical systems. volume 2478, pages 74–91.

Vallejo, D., Castro-Schez, J. J., Glez-Morcillo, C., and Albusac, J. (2020). Multi-agent architecture for information retrieval and intelligent monitoring by UAVs in known environments affected by catastrophes. *Engineering Applications of Artificial Intelligence*, 87:103243.