FIDES: A Neuro-Symbolic Conversational Tool for Faithful Production Process Intelligence

Angelo Casciani[1], Fabrizio Italia[1], Livia Lestingi[2], Matteo Marinacci[1], Andrea Marrella[1], and Andrea Matta[2]

[1] Sapienza University of Rome, Rome, Italy,
lastname@diag.uniroma1.it
[2] Polytechnic University of Milan, Milan, Italy,
firstname.lastname@polimi.it

Abstract. Modern production systems demand timely diagnostic and prognostic insights, yet the complexity of existing process intelligence (PI) tools creates a high technical barrier even for domain experts. This paper presents FIDES, a conversational neuro-symbolic tool designed to enable access to these analysis engines via natural language. Unlike approaches uniquely based on Large Language Models (LLMs) that often hallucinate operational results, FIDES implements a sound routing architecture: it uses LLMs strictly for understanding and translating the user's intent into machine-readable encodings, while delegating the orchestration to a domain-independent automated planner. The planner autonomously decomposes complex queries and routes them to the appropriate PI engines, ensuring rigorous results. We demonstrate the tool's maturity and usability through a lab-scale manufacturing case study, highlighting how its web-based interface enables non-technical users to perform faithful multi-perspective analysis of production processes.

Keywords: Production System, Large Language Model, Simulation, Formal Verification, Process Mining, Process Intelligence

1 Introduction and Problem Description

With the advent of Industry 4.0, production systems are experiencing an important digital transformation. Internet-of-Things devices now generate vast amounts of *production data*, enabling the creation of *digital models* that reflect manufacturing activities throughout their lifecycle [15]. Manufacturing companies aim to leverage this data to augment human expertise, fostering a collaborative environment where decision-making is faster, more accurate, and informed [2].

However, achieving this goal is hindered by significant barriers. Current process intelligence (PI) tools (e.g., for simulation, verification, process mining) operate in silos and require highly skilled users to configure them and interpret raw outputs. Most domain experts (e.g., plant managers) lack the technical expertise to interact with these tools effectively. Consequently, decision-making often relies on manual data selection, which is time-consuming, error-prone, and limited to single-perspective analyses, increasing the risk of sub-optimal decisions [10].

Furthermore, the inadequacy of classical information systems engineering paradigms in managing the volatility of modern digital ecosystems calls for new methodologies and tools [16]. Sudden machine breakdowns, urgent order injections, and fluctuating resource availability demonstrate this volatility in the manufacturing domain [7]. Traditional PI systems rely on static dashboards and rigid workflows designed for stable requirements [16]. When a change occurs, the engineering effort required to reconfigure a simulation model or verify a new property is often too time-consuming to support real-time decision-making.

To address these challenges, we previously introduced a conceptual conversational framework for production systems [6]. In this paper, we extend that work with a concrete implementation by presenting FIDES, a neuro-symbolic conversational tool for PI designed to understand the user's intent, translate it into machine-readable instructions, and orchestrate PI tools. In particular, FIDES adopts an *"LLM-as-formalizer"* approach [20] from the ground up. Our architectural philosophy draws a direct analogy to the *dual-process theory* proposed by Kahneman [9]. We treat the Large Language Model (LLM) strictly as a *"system 1"* translator to handle the ambiguity of natural language. Conversely, we treat the PI engines as *"system 2"* components for the sound computation of rigorous results.

Unlike our initial work, FIDES features a user-friendly graphical user interface (GUI) and introduces a novel *sound routing mechanism* to orchestrate this duality. By replacing rigid graphical controls with a neuro-symbolic conversational interface, we enable domain experts to query the underlying digital models via natural language. While LLMs are good at language processing and translation, they are unreliable planners [17]. To prevent the "system 1" LLM from hallucinating operational outcomes, FIDES integrates a "system 2" Artificial Intelligence (AI) planner, ensuring that complex user requests are decomposed into a sound sequence of operations before being delegated to the appropriate PI engines, specifically a *simulation*, a *formal verification*, and a *process mining* module.

The tool is designed for *non-technical users* who need *actionable* insights without having deep technical knowledge of the PI engines [13]. A typical scenario involves a plant manager asking: "*What is the impact of station A failure on waiting times for producing x pieces?*" FIDES autonomously finds a sequence of actions needed to compute the requested information and routes the encoded tasks to the simulation engine. Then, it returns a faithful natural language answer, abstracting away the underlying results from the engine.

The novelty of this demonstration lies in this modular, *neuro-symbolic* software architecture. By combining the linguistic flexibility of LLMs with the reliability of formal solvers, we address the reasoning limitation of LLMs, ensuring that the chat interface does not compromise the validity of the results. We demonstrate the tool's maturity through a lab-scale case study within the MOTOWN project[1], highlighting its reliability and usability.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes the software architecture, Section 4 presents the demonstration

[1] MOTOWN project: `http://www.open.diag.uniroma1.it/project_detail/28097`

scenario, and Section 5 assesses the maturity of the tool. Finally, Section 6 reports the tool's current limitations and Section 7 concludes the paper.

2 Related Work

The application of LLMs in production systems is a rapidly growing field. Recent tools have integrated LLMs to automate simulation parameterization [22], optimize energy efficiency [21], or serve as interfaces for digital twins [18].

In the context of LLMs for automated planning, existing solutions generally fall into two categories [20]. *LLM-as-planner* approaches use the LLM directly to generate plans, suffering from hallucinations and failing to account for operational constraints [17], which is unacceptable in safety-critical industrial settings. Conversely, *LLM-as-formalizer* techniques couple LLMs with specific solvers (e.g., uniquely for heat treatment [19] or for verification [3]). However, these lack a unified orchestrator capable of handling composed PI requests that require multi-perspective analysis, such as combining process mining (for process discovery) with simulation (for what-if analysis).

FIDES differentiates itself by adopting an *LLM-as-formalizer* architecture [20]. Unlike purely LLM-based tools, we do not ask the LLM to solve the problem, but we use it strictly as a "system 1" translator to convert the natural language user's intent into formal encodings. A sound, domain-independent planner, acting as "system 2", then decomposes these queries and routes them to the appropriate PI engine. This hybrid approach ensures that the flexibility of natural language is backed by the sound guarantees of formal solvers, supporting reliability and usability.

3 Tool and Software Architecture

The architecture of FIDES follows a *modular*, *two-tier* design divided into *frontend* and *backend*, as illustrated in Figure 1, and it is implemented in Python (`v3.11`), ensuring interoperability between the components. The starting point for the tool is an event log in XES format capturing the historical execution traces of the production system, grounding the analysis in empirical data [1]. To facilitate reproducibility and deployment, the entire tool is containerized using Docker[2].

Frontend. The entry point for the user is the Web-based GUI, built using the Gradio[3] library, which provides a responsive chat UI suitable for prototyping and demonstration. This interface enables non-technical stakeholders, such as plant managers, to input natural language queries (e.g., *"simulate the production of 200 pieces assuming a breakdown on station A"*). The frontend also handles the visualization of the answer, presenting both textual summaries of the PI tasks execution and artifacts (e.g., process model discovered from an event log of the production process) returned by the backend.

[2] Docker documentation: `https://www.docker.com/`
[3] Gradio documentation: `https://www.gradio.app/`

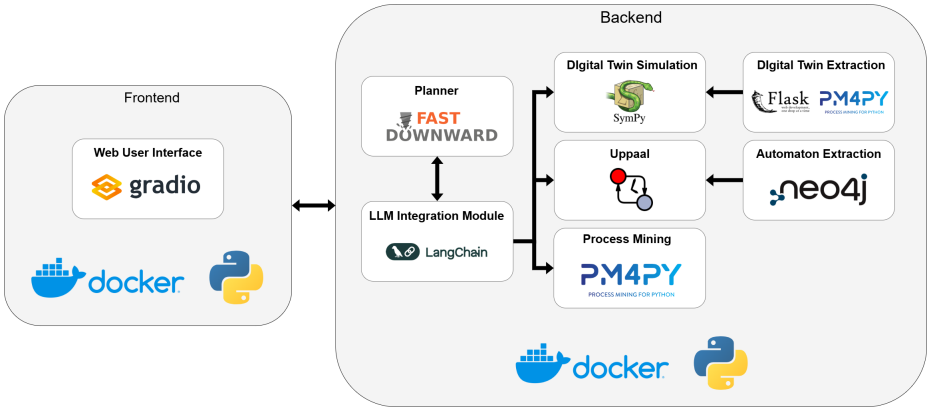

Fig. 1. High-level software architecture of the FIDES.

Backend. The core of the tool, responsible for the "neuro-symbolic" routing, encompasses two main components. The *LLM Integration Module* acts as the semantic parser, and it is implemented with LangChain[4], supporting both local and API-based language models. Instead of answering the user directly, it translates the natural language intent into a formal goal specification. Simple requests are routed directly to the appropriate PI engine. Conversely, requests composed of multiple goals are formalized as *Planning Domain Definition Language* (PDDL) problems and dispatched to a domain-independent *planner*, ensuring the system does not hallucinate execution outcomes. For this, we employ Fast Downward [8], which receives the goals and generates a sound plan that defines which engines are required to solve the problem (e.g., *discover model* $\rightarrow$ *verify property* $\rightarrow$ *run simulation*).

The backend also hosts the specialized PI engines that perform the actual analysis tasks. As the input of FIDES is the raw event log in XES, the architecture integrates two extraction components to derive up-to-date digital models directly from execution data, ensuring they accurately reflect the current state of the production system. The PI tools then operate either on these extracted artifacts or directly on the event log.

In particular, the *Process Mining* module uses PM4Py to parse logs and perform process discovery, conformance checking, variant analysis, and filtering. The *Simulation* engine, built using SimPy[5], runs discrete event simulations to compute performance metrics (e.g., throughput, waiting times) based on the parametrized digital twin and the scenarios configured by the orchestrator according to the user's request. Finally, the verification module uses UPPAAL [11] for checking safety properties in Timed Computation Tree Logic (TCTL) (e.g., deadlock freedom) against the extracted Probabilistic Timed Automata (PTA) [5].

The *Digital Twin Extraction* module, implemented as a Flask service, uses PM4Py[6] to compute process model and simulation parameters directly from XES

[4] LangChain documentation: `https://www.langchain.com/`
[5] SimPy documentation: `https://simpy.readthedocs.io`
[6] PM4Py documentation: `https://processintelligence.solutions/pm4py`

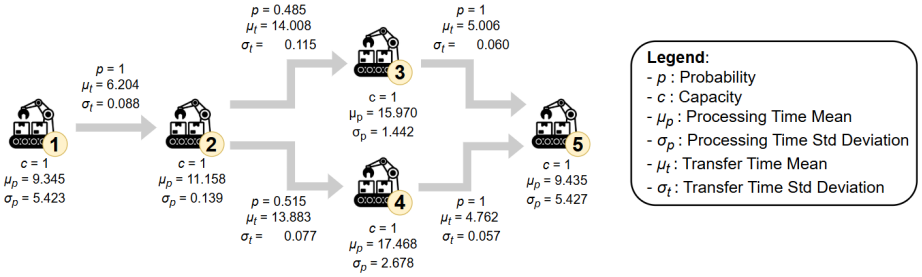

Fig. 2. The LEGO factory annotated with the corresponding simulation parameters extracted from the event log. Time-related parameters are in seconds (s).

event logs via split miner [4]. Similarly, the *Automaton Extraction* engine uses Python and Neo4j[7] to maintain a system knowledge graph for verification and implements the L^*_{sha} algorithm to learn stochastic hybrid automata from collected system traces, providing the formal models required for safety checks [12].

4 Demonstration

To demonstrate the capabilities of FIDES, we utilize a lab-scale reproduction of a real-world manufacturing line, i.e, the *Lego factory* developed within the MOTOWN project. The physical twin (see Figure 2) consists of multiple processing stations connected by conveyor belts and its execution is captured in a XES event log. A common operational challenge in this setting is managing the *trade-off* between *throughput* and *safety* (e.g., deadlock freedom) when production targets change. In this scenario, let us assume a plant manager who needs to validate a production batch of 20 units but, importantly, they must first ensure that the current configuration of the line is safe from stalling.

Figure 3 shows an interaction with the GUI, which proceeds as follows.

Step 1: Natural Language Request. The user inputs a multi-objective query into the chat interface: *"Verify if the production process is deadlock-free and tell me how much time is needed to produce 20 pieces."* This request is challenging as it requires a safety check (which needs verification from UPPAAL) and a quantitative performance estimation (via simulation by the digital twin module).

Step 2: Neuro-Symbolic Orchestration. Instead of attempting to answer directly, the LLM Integration Module parses the intent and forwards the generated PDDL problem to the Fast Downward planner. The planner analyzes the request and generates a sound two-step execution plan, including the deadlock check and the simulation of the production of 20 units.

[7] Neo4j documentation: `https://neo4j.com/docs/`

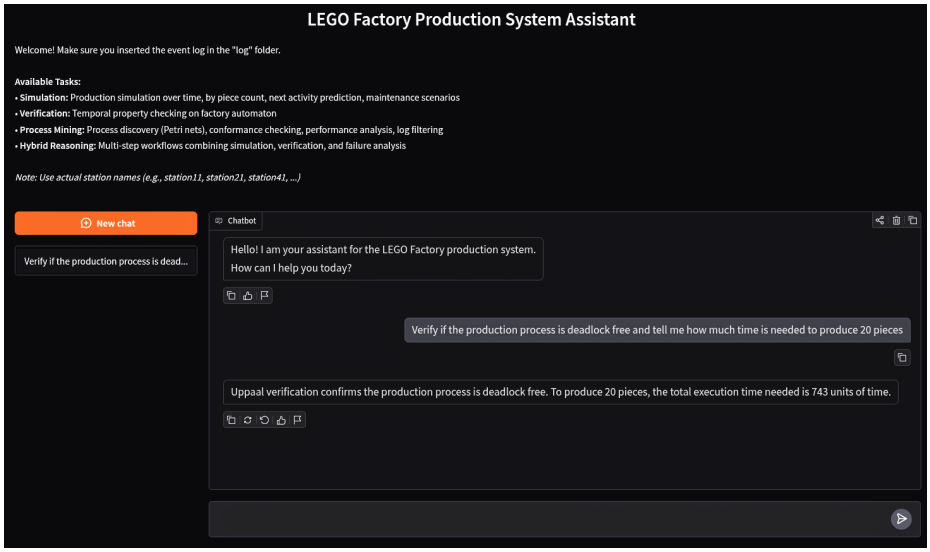

Fig. 3. Demonstration of the FIDES user interface.

Step 3: Execution and Artifact Generation. Next, FIDES executes the plan sequentially. First, UPPAAL checks for the absence of a deadlock against the PTA and returns a result (e.g., *"property satisfied"*). Then, the simulation module initializes the digital twin with the specified parameters (20 pieces in this case) and runs the discrete event simulation, returning a value (e.g., 45 minutes).

Step 4: Response Generation. Ultimately, the LLM aggregates the raw outputs from both engines, constructing a natural language response that contextualizes the technical results for the user, as shown in the chat window in Figure 3.

5 Maturity and Availability

Maturity. FIDES is currently a research prototype at *Technology Readiness Level* (TRL) 4, i.e., a technology validated in a laboratory environment [14], developed within the MOTOWN national research project. The system is built upon mature, industry-standard libraries and tools, including SimPy, UPPAAL, PM4Py, and Fast Downward.

To validate the requests orchestration, we conducted a *quantitative evaluation* assessing the tool's ability to route and encode requests across varying complexity levels, ranging from simple simulation tasks to hybrid optimization. The results for a representative subset of models are summarized in Table 1. Our benchmarks confirm that the orchestration layer achieves near-perfect routing accuracy ($\geq$98%) when employing state-of-the-art proprietary models (e.g., `gpt-5.1` or `gemini-2.5-flash`). While open-weight models like `mistral-nemo` show promise in encoding specific queries, smaller models occasionally report hallucinated execution results instead of responding with the required encoding. This explicitly justifies the need

Table 1. Few-shot accuracy over five runs for *routing* (R) and encoding of *simulation* (SI), *verification* (V), *process mining* (PM), *structural information* (ST), and *hybrid* (H) tasks.

Model	*R*	*SI*	*V*	*PM*	*ST*	*H*
llama-3.2-1b	0.21	0.48	0.02	0.52	0.81	0.08
llama-3-8b	0.70	0.79	0.78	1.00	0.98	0.52
mistral-nemo	0.91	0.92	0.72	1.00	1.00	0.63
gemini-2.5-flash	**0.99**	0.99	**0.99**	**1.00**	0.92	0.95
gpt-5.1	0.98	**1.00**	0.90	**1.00**	**1.00**	**0.99**

for a tool like FIDES, which uses the LLM solely for translation and offloads the task execution to deterministic PI engines.

Furthermore, our experiments indicate that medium-sized open-source models perform reliably on atomic encoding tasks, suggesting that the tool can support a privacy-preserving mixed deployment. Organizations can use a powerful LLM for the routing logic while delegating data-sensitive schema encoding to local models. This is a critical advantage for manufacturing domains, as it allows them to maintain full control over their production data and mitigate the risks associated with external API calls.

Moreover, a group of academics and graduate students assessed the conversational tool with respect to usability. Their feedback highlighted the system's ability to correctly interpret requests even when they were not perfectly formalized, suggesting that the natural language interface can lower the PI barrier to entry compared to configuring traditional dashboards.

Availability. The tool is released as an open-source project to foster collaboration and transparency. The source code is available on GitHub at: `https://github.com/angelo-casciani/FIDES`. A video demonstration showcasing the tool in the Lego factory use case is available at: `https://youtu.be/bZUNWQW07bc`.

6 Limitations and Threats to Validity

While FIDES lowers the barrier to entry for PI, it is not a universal solution, and its effectiveness is bounded by several limitations. First, regarding *reliability and traceability*, the multi-hop neuro-symbolic architecture introduces several layers of abstraction. While the planner guarantees that the orchestration is *syntactically* valid and logically sound, ensuring *semantic* alignment with the user's goal remains challenging. As natural language is inherently ambiguous, the LLM might interpret the request in a way that does not completely align with the user's true intent. Furthermore, abstracting the PI engine complexity away from the user naturally reduces traceability: the human receives an answer but cannot easily verify the intermediate encodings, a challenge already common to many commercial PI platforms. Additionally, the sensitivity of the final PI results to minor variations in the user's initial prompt formulation requires further investigation.

Second, regarding *system requirements and scope*, the tool currently requires substantial manual engineering to define the planning domain in PDDL and the specific preconditions and effects for each integrated PI tool. As a result, FIDES is effective at orchestrating a bounded set of pre-configured tools but cannot yet automatically integrate unseen engines out-of-the-box.

Finally, we acknowledge *threats to validity* deriving from the rapid evolution of LLMs, which could alter the performance baselines. To mitigate this, our preliminary evaluation considered both proprietary and open-source models, demonstrating that the architectural benefits hold regardless of the underlying LLM. To address the inherent stochasticity of LLMs, we averaged performance over five runs and provided the employed prompts to ensure reproducibility.

7 Conclusion

This paper presented FIDES, a conversational tool based on neuro-symbolic AI technologies that democratizes access to faithful production PI. By bridging the gap between natural language and sound PI engines, we empower non-technical users to perform multi-perspective analysis tasks, from process mining and production simulations to safety verifications, without needing to master the specific syntax of each tool. Aligning with Kahneman's dual-process theory, the main innovation of our architecture is the use of an automated planner as a *"system 2"* component for sound routing of requests encompassing the invocation of multiple PI engines, mitigating the risk of LLM (i.e., *"system 1"*) hallucinations in critical decision-making for production processes.

Future work will focus on enhancing the framework's autonomy, extensibility, and usability. To improve autonomy, we aim to implement a self-correcting feedback loop in which the LLM uses error messages from the PI tools to iteratively refine its translations without human intervention. To support extensibility, we plan to automate the formalization of the planning domain by automatically extracting the preconditions and effects of new PI engines, avoiding the current manual PDDL encoding. Regarding validation, we will move beyond the lab-scale case study to test the tool against large-scale industrial logs, identifying potential bottlenecks in the orchestration layer. Finally, we intend to develop a low-code graphical interface to enhance accessibility, featuring pre-structured question patterns to assist users with limited technical backgrounds in formulating semantically unambiguous queries.

Acknowledgments. This work was supported by the Sapienza project FOND-AIBPM, the PRIN 2022 project MOTOWN, the PNRR MUR project PE0000013-FAIR, and the EU Horizon Europe Research and Innovation Programme under Grant 101092021 (AutoTwin). Angelo Casciani conducted this research while enrolled in the Italian National Doctorate in Artificial Intelligence run by Sapienza University of Rome (CUP B53C23003500006, Mission 4 NRRP Generic Research Scholarship, Component 1). Matteo Marinacci is supported by Thales Alenia Space and Regione Lazio, through the fellowship 35752-22066DP000000040-A0627S0030 *Utilizzo dell'Intelligenza Artificiale a supporto della progettazione e qualifica del prodotto spaziale.*

References

1. Acampora, G., Vitiello, A., Stefano, B.N.D., van der Aalst, W.M.P., Günther, C.W., Verbeek, E.: IEEE 1849: The XES standard. IEEE Comput. Intell. Mag. **12**(2), 4–8 (2017)
2. Agostinelli, S., Benvenuti, D., Casciani, A., Luzi, F.D., Marinacci, M., Marrella, A., Rossi, J.: A Context-Aware Framework to Support Decision-Making in Production Planning. In: CAiSE 2024. pp. 248–264 (2024)
3. Arora, D., Kambhampati, S.: Learning and leveraging verifiers to improve planning capabilities of pre-trained language models. CoRR **abs/2305.17077** (2023)
4. Augusto, A., Carmona, J., Verbeek, E.: Advanced process discovery techniques. In: Process Mining Handbook, LNBIP, vol. 448, pp. 76–107. Springer (2022)
5. Beauquier, D.: On probabilistic timed automata. Theoretical Computer Science **292**(1), 65–84 (2003)
6. Casciani, A., Lestingi, L., Marrella, A., Matta, A.: A conversational framework for faithful multi-perspective analysis of production systems. In: CAiSE 2025. LNCS, vol. 15701, pp. 163–181. Springer (2025)
7. Frigerio, N., Tan, B., Matta, A.: Simultaneous control of multiple machines for energy efficiency: a simulation-based approach. Int. J. Prod. Res. **62**(3), 933–948 (2024)
8. Helmert, M.: The fast downward planning system. J. Artif. Intell. Res. **26**, 191–246 (2006)
9. Kahneman, D.: Thinking, fast and slow. Penguin, London (2012)
10. Keskin, Z., Joosten, D., Klasen, N., Huber, M., Liu, C., Drescher, B., Schmitt, R.H.: LLM-Enhanced Human–Machine Interaction for Adaptive Decision-Making in Dynamic Manufacturing Process Environments. IEEE Access **13**, 44650–44661 (2025)
11. Larsen, K.G., Pettersson, P., Yi, W.: UPPAAL in a nutshell. Int. J. Softw. Tools Technol. Transf. **1**(1-2), 134–152 (1997)
12. Lestingi, L., Sbrolli, C., Scarmozzino, P., Romeo, G., Bersani, M.M., Rossi, M.: Formal modeling and verification of multi-robot interactive scenarios in service settings. In: FormaliSE@ICSE 2022. pp. 80–90. ACM (2022)
13. Liu, M.X., Liu, F., Fiannaca, A.J., Koo, T., Dixon, L., Terry, M., Cai, C.J.: "We Need Structured Output": Towards User-centered Constraints on Large Language Model Output. In: Ext. Abstracts of the CHI Conf. on Human Factors in Computing Systems. pp. 10:1–10:9. ACM (2024)
14. Mankins, J.C., et al.: Technology readiness levels (1995)
15. Negri, E., Berardi, S., Fumagalli, L., Macchi, M.: MES-integrated Digital Twin Frameworks. J. of Manufacturing Syst. **56**, 58–71 (2020)
16. Porra, J., Hirschheim, R., Land, F., Lyytinen, K.: Seventy years of information systems development methodologies from early business computing to the agile era: A two-part history. part 1: From pre to early isd methodology era: The emergence of isd methodologies and their golden era (1880–1980). Journal of Information Technology **40**(4), 441–469 (2025)
17. Stechly, K., Valmeekam, K., Kambhampati, S.: On the self-verification limitations of large language models on reasoning and planning tasks. In: ICLR 2025. OpenReview.net (2025)
18. Sun, Y., Zhang, Q., Bao, J., Lu, Y., Liu, S.: Empowering digital twins with large language models for global temporal feature learning. J. of Manufacturing Syst. **74**, 83–99 (2024)
19. Sun, Y., Li, X., Liu, C., Deng, X., Zhang, W., Wang, J., Zhang, Z., Wen, T., Song, T., Ju, D.: Development of an intelligent design and simulation aid system for heat treatment processes based on llm. Materials & Design **248**, 113506 (2024)
20. Tantakoun, M., Muise, C., Zhu, X.: Llms as planning formalizers: A survey for leveraging large language models to construct automated planning models. In: Findings of the ACL 2025. pp. 25167–25188. ACL (2025)

21. Wang, Z., Qin, H.: Intelligent industrial production process automatic regulation system based on llm agents. In: AIEA 2024. pp. 133–137. IEEE (2024)
22. Xia, Y., Dittler, D., Jazdi, N., Chen, H., Weyrich, M.: Llm experiments with simulation: Large language model multi-agent system for simulation model parametrization in digital twins. In: ETFA 2024. pp. 1–4 (2024)