Leveraging LLMs and RAG for Data Service Discovery in the Internet of Production

(Discussion Paper)

Devis Bianchini[1], Valeria De Antonellis[1], Massimiliano Garda[1,*], Michele Melchiori[1] and Anisa Rula[1]

[1]*University of Brescia, Dept. of Information Engineering*
Via Branze 38, 25123 - Brescia (Italy)

Abstract

The Internet of Services paradigm enables data-driven collaboration in production networks, but its full potential is hindered by the gap between domain experts and R&D managers, who define data needs, and IT specialists, who integrate data services into analytics pipelines. Large Language Models (LLMs) have recently emerged as valuable tools to assist domain experts in identifying and combining data services into pipelines, thus bridging the gap with IT specialists. However, building effective prompts for LLM-based systems requires technical expertise and extensive knowledge of available services and their integration. To address this aspect, we propose an LLM-based approach for data service discovery in the Internet of Production, leveraging Retrieval-Augmented Generation (RAG) applied to a catalog of atomic data services and analytics pipelines. This approach helps domain experts and R&D managers specify service needs and draft preliminary analytics pipelines, which IT specialists can then implement. The evaluation conducted in real-world Smart Factory case study demonstrates its potential to streamline data service discovery and analytics pipelines design.

Keywords

Large Language Models, Retrieval-Augmented Generation, prompt engineering, Internet of Services, data services

1. Introduction

The Internet of Services (IoS) paradigm utilizes data services to support various stages of data analytics (e.g., storage, processing, analysis) in Smart Manufacturing processes. These services can be discovered and combined into data analytics pipelines, improving manufacturing efficiency and fostering collaboration among actors like domain experts, R&D managers, and IT specialists [1]. While domain experts and R&D managers know which data should be collected, IT specialists oversee the discovery and integration of data services into pipelines. Traditional approaches rely on formal descriptions and annotations to describe the internal and external behaviour of services, requiring significant technical expertise. Recently, Large Language Models (LLMs) have emerged as a tool for service discovery, providing a support to bridge the gap between domain experts and IT specialists [2]. LLMs enable domain experts to

SEBD 2025: 33rd Symposium on Advanced Database System, June 16–19, 2025, Ischia, Italy
*Corresponding author.
devis.bianchini@unibs.it (D. Bianchini); valeria.deantonellis@unibs.it (V. De Antonellis); massimiliano.garda@unibs.it (M. Garda); michele.melchiori@unibs.it (M. Melchiori); anisa.rula@unibs.it (A. Rula)

preliminarily select services and design analytics pipelines, with IT specialists finalizing the implementation. However, interacting effectively with LLMs remains a challenge, particularly in coping with the *hallucination* phenomenon, which compels users to properly formulate prompts reflecting their search intent [3]. Although prompt engineering aims to address the latter issue, it still depends on human expertise to build precise prompts. To improve service discovery, Retrieval-Augmented Generation (RAG) techniques can be employed [4], but current literature uses limited service models that overlook the inclusion of analysis scenarios as relevant examples to guide the design of new pipelines [5, 6, 7]. This paper proposes an approach for data service discovery in Internet of Production contexts, leveraging LLMs. The approach includes: (i) designing prompt templates to help domain experts and R&D managers formulate search intents, and (ii) using RAG on a catalog of atomic data services and analysis scenarios. Preliminary experiments have been conducted in a real-world case study within the scope of the MICS national research project [8], funded by the NextGenerationEU economic recovery package. This discussion paper stems from our previous paper [9], where we provided an overview of the requirements and the building blocks for achieving LLM-based data services discovery. The remainder of the paper is organised as follows. In Section 2 related work is discussed. Section 3 introduces the case study and background concepts. Section 4 provides details on the ingredients for LLM-based service discovery, as well as a preliminary evaluation of the approach. Finally, Section 5 closes the paper.

2. Related work

Recent studies have explored the use of LLMs for service discovery across various application domains. Zeng et al. [7] propose using LLMs for automatic workflow generation involving APIs, with a structured prompt to guide the search process. Huang et al. [10] integrate LLMs with a Knowledge Graph for API recommendation, overcoming challenges like rigid templates and out-of-vocabulary issues. Monti et al. [6] introduce NL2ProcessOps, which uses LLMs and RAG for code generation to support process deployment. Li et al. [5] combine ChatGPT and a Manufacturing Service Knowledge Graph to address domain-specific queries in manufacturing service discovery. Nonetheless, these approaches lack a consideration of how to combine RAG methods with structured prompt templates incorporating service descriptive metadata. Indeed, Zeng et al. templates focus mainly on output style rather than structuring users' inputs through placeholders, while Huang et al. provide details about service descriptive metadata only at a high level of abstraction. The Knowledge Graph employed by Li et al. only provides basic textual service descriptions as metadata, and templates from Monti et al. prioritize code generation over the employment of service metadata for prompt design. Conversely, our LLM-based approach emphasizes the usage of terminology and concepts descending from analysis scenarios and services descriptive metadata to design prompt templates to interact with an LLM-based system for service discovery. Moreover, the approach fosters the RAG method to enhance LLM contextual knowledge about data services and meaningful analytics pipelines.

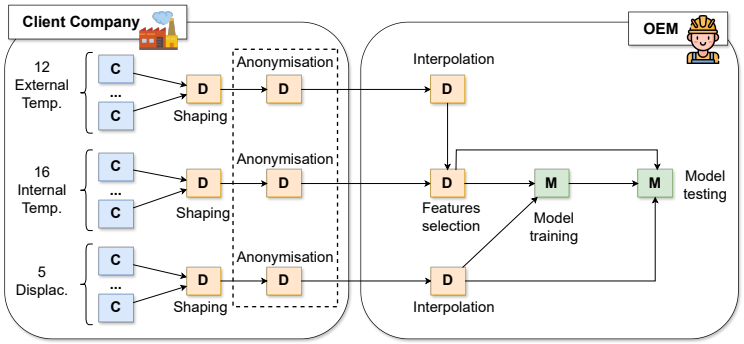

Figure 1: Pipeline of ADS for the example scenario (legend for service types – C: COLLECT, D: DISPATCH, M: MONITOR).

3. Case study and background concepts

Let us consider the perspective of an Original Equipment Manufacturer (OEM), aiming to monitor the thermal deformation (*thermal error*) of a five-axis vertical machine used for milling complex three-dimensional parts in steel, aluminium alloys, and composites. In milling machines, the thermal error is one of the most significant factors leading to a structural deformation (observed between the work-piece and the tool acting on the work-piece), influencing the accuracy of the machine tool and decreasing product quality. One of the tests apt to assess the magnitude of thermal error is the *axis-motion test*, separating the contributes of thermal displacement by assuming that each single axis shows a different behaviour and, therefore, has to be moved and tested separately. In this respect, thermal error is ascribable to the displacements $d_{i=1...5}$, where i refers either to a Cartesian or to a rotational axis, measured from the zero-displacement position. Regression models (such as MLRA or LASSO), can be employed to predict displacements $d_{i=1...5}$ as a function of temperatures gauged through sensors mounted by the OEM both internally and externally the machine. In fact, in operational settings, a direct measure of displacements is not feasible. The validated prediction model can be employed during daily machine operation, when only temperature measurements are available and displacements cannot be detected due to the presence of the work-piece and material shavings.

Atomic Data Services and Analysis Scenarios. The data analysis procedure to accomplish error predictions is conceivable as a sequence of several atomic tasks (e.g., the measure of temperature through sensors, the selection of train/test data for regression models). These analytical tasks are widely applicable across diverse Smart Manufacturing domains, extending beyond the discussed case study. For the execution of these tasks, according to an IoS paradigm, we rely on a catalog of *Atomic Data Services* (in brief, Data Services or ADS) which, depending on their functionalities, are distinguished amongst: (i) COLLECT services, used by actors within a production network to retrieve data (e.g., from sensors, data stores); (ii) DISPATCH services, to manipulate data and share it across the production network; (iii) MONITOR services, to perform analysis tasks such as prediction or data comparison. Pipelines of ADS are constructed to

accomplish *data analysis scenarios* (in brief, analysis scenarios). An example of a data analysis scenario, taken from the considered case study, is when the OEM performs data analysis based on measures collected from a milling machine hosted by one of his/her clients. Here, data analysis is offered as-a-Service to the client. A pipeline of ADS fulfilling this scenario is shown in Figure 1. Other scenarios, implemented as pipelines of ADS, can be identified in a similar way, in the same Smart Factory context or in different ones, as well.

Atomic Data Services and Analysis Scenarios modelling. The informative elements apt to model ADS and Analysis Scenarios, serving as descriptive metadata, are briefly summarised in the following and highlighted in small-caps writing style. Each ADS processes DATASETS containing DATA RECORDS, abstracting data regardless of type. MONITOR services are organized among CLASSIFICATION services (producing class labels), PREDICTION services (generating continuous values), and COMPARISON (evaluating their inputs against thresholds). ADS data operations are modelled in a declarative way, resembling the primitives of a declarative language (e.g., SQL): PROJECTION (attribute selection), SELECTION (data filtering), AGGREGATION (applying functions like MIN/MAX/AVG), and TRANSFORMATION (e.g., format conversion). Each service is associated with a DESCRIPTION DOCUMENT, gathering ADS metadata, following standards like OpenAPI [11]. Instead, an ANALYSIS SCENARIO is modelled as a pipeline of ordered STEPS (e.g., data collection) supervised by ACTORS (e.g., client, OEM), where each step comprises TASKS (e.g., internal temperatures collection) executed by specific ADS. SPECIALIZATION relationships between scenarios can be established to reflect that one scenario adds steps/tasks to another.

4. LLM-based system for data service discovery

Prompt templates to interact with the LLM-based system. In the following, we present prompt templates (Figure 2) designed for the LLM-based service discovery system, relying upon the terminology and concepts related to ADS and analysis scenarios descriptive metadata. Specifically, at this stage of the research, we devised three prompt templates (namely, T1, T2 and T3) pursuing different objectives and built complying with renowned organizational patterns [12]. Instances of such prompts may be chained together to assure a continuous interaction flow, where the output obtained by the LLM-based system exploiting a prompt can be used as (part of) the input for the subsequent one. This technique is denoted as *prompt chaining* [13], and it is useful to relieve the LLM at query time, as it may struggle answering it. In each template, *placeholders* for the user's input are delimited by curly brackets (e.g., `{task}`) and derive from the descriptive metadata of ADS and analysis scenarios, except for the `context` placeholder. Unlike the others, in fact, `context` is not intended to host any textual input from the user; instead, it serves as a special reference for passing in formatted documents to the prompt. These are retrieved through the RAG method as described in the next paragraph. Prompt templates can be employed to pursue three *interaction strategies*, targeted to domain experts and R&D managers: (i) assist users without a clear idea of the structure of the analysis scenario, by using templates T1-T3 to progressively define steps, tasks, and discover appropriate ADS; (ii) help users with an already defined scenario, by directly utilizing template T3 to discover relevant ADS; (iii) focus solely on scenario structured through templates T1-T2,

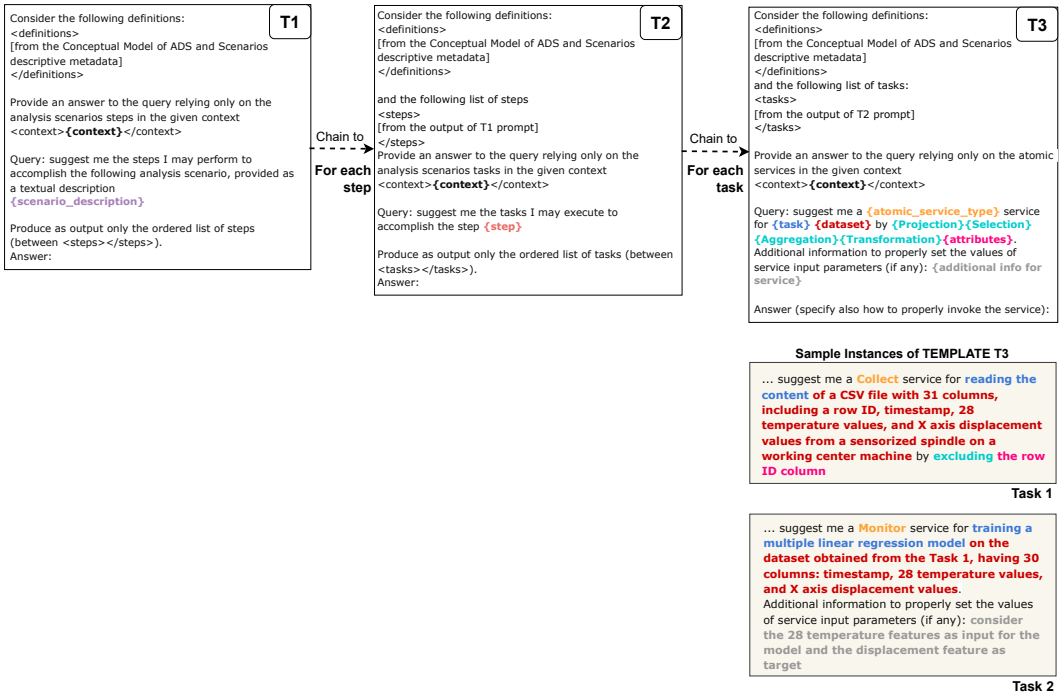

Figure 2: Prompt templates T1 to T3. Example of instances of template T3.

postponing ADS discovery.

Retrieval-Augmented Generation method. The proposed LLM-based system for ADS discovery leverages the Retrieval-Augmented Generation (RAG) method [4] to enhance an LLM with contextual data, thereby improving response coherence and relevance to users' queries. In our case, contextual data has a two-fold nature, as it regards: (i) a catalog of ADS, described according to the OpenAPI specification standard (in JSON format), which enables the inclusion of descriptive metadata based on the ADS model; (ii) descriptions of available analysis scenarios (in XML format), according to the data analysis scenarios model. From a higher viewpoint, the RAG method is composed of three main macro-phases: (a) *indexing*, by applying a chunking strategy (to ensure efficient storage and faster retrieval), and an indexing strategy to the descriptions of ADS and analysis scenarios to be stored into a vector database as numeric vectors (exploiting an *embedding model*); (b) *retrieval*, which takes the user's textual query and retrieves the relevant data chunks from the vector database using the query; (c) *generation*, wherein a prompt containing the original user's query and contextual knowledge from the vector database is processed by the LLM to provide an answer to the user. The LLM-based system equipped with the RAG method has been implemented with LangChain [14] and hosted on Google Colab notebooks. These notebooks run the Open Source *granite3-dense-8B* LLM, utilizing Google Cloud Platform GPUs, and implement the RAG method over OpenAPI and analysis scenario datasets. The vector database for the RAG module em-

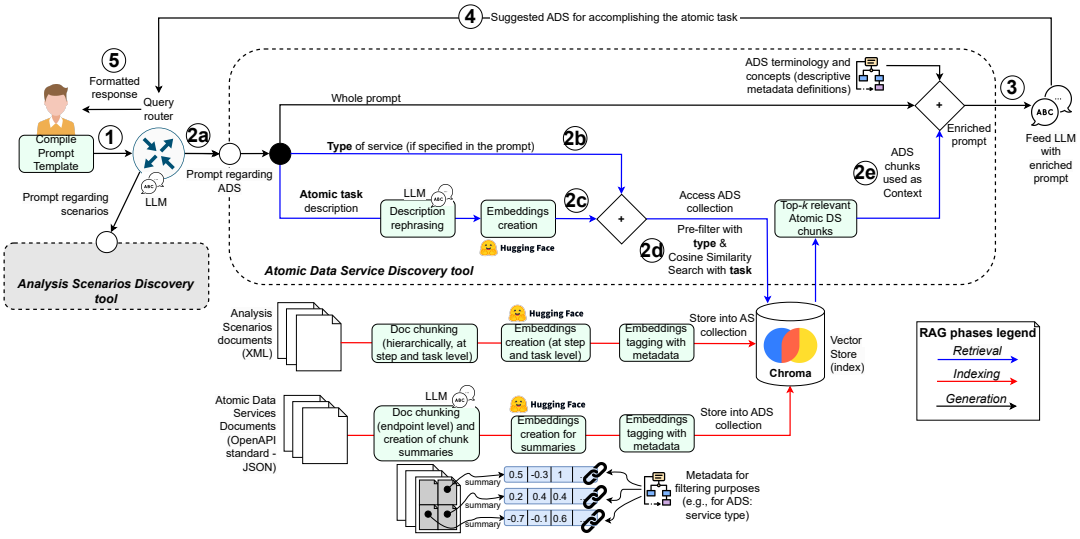

Figure 3: Overview of RAG method: details of the ADS discovery process with prompt template T3.

ploys ChromaDB [15], storing embeddings of ADS and analysis scenarios in separate collections.

Example of RAG-based discovery workflow. Figure 3 illustrates the workflow of the ADS discovery process using prompt template T3, with circled numbers indicating the order of operations. The user-specified placeholders in T3 are employed during the Retrieval and Generation phases of RAG (2a). In particular, in the Retrieval phase, the placeholder `{atomic_service_type}` is used as a filtering key (2b), while a rephrased version of `{task}` (2c) is employed to perform a cosine similarity search, restricted to a subset of ADS document chunks (2d). In the Generation phase, the retrieved ADS chunks (2e) provide context for LLM inference (3), allowing the model to suggest the appropriate ADS to accomplish the atomic task (4, 5).

Preliminary validation of the LLM-based system with the proposed prompt templates. Leveraging prompt templates T1, T2 and T3, we conducted a qualitative evaluation to assess whether LLM-generated responses downstream the Generation phase of RAG align with user's requests. To this aim, we recruited a group of seven users from the application context presented in Section 3, knowledgeable of LLMs and prompts usage, having the skills of domain experts and R&D managers. The evaluation applied *In-Context Learning* (ICL) techniques [21] over three groups of prompts, designed in turn with three prompting strategies: (i) *zero-shot* (templates T1-T3), (ii) *one-shot* (templates T3), and (iii) *Chain-of-Thought* (templates T1-T2). Two LLM configurations were tested: (i) full OpenAPI documents descriptions and (ii) partial OpenAPI documents descriptions (omitting meaningful names in service routes, parameters, etc.). In the following, we report some evidences from the conversations between users and the LLM-based system: (i) non-template queries required three interactions for relevant ADS retrieval; (ii) zero-shot ICL effectively identified, normally at the first interaction, tasks and ADS with full OpenAPI

documents descriptions; (iii) partial OpenAPI descriptions induced minor hallucinations (e.g., non-existent services). The planned future work on broadening the experimental evaluation includes a quantitative assessment of both Retrieval and Generation phases of RAG as well as the optimization of RAG hyperparameters (e.g., related to the chunking strategy).

5. Conclusions and future work

In this paper, we proposed an LLM-based approach for data services discovery in the Internet of Production context, leveraging Retrieval-Augmented Generation (RAG) on a catalog of atomic data services and analysis scenarios. Our approach emphasizes the use of tailored prompt templates to guide user's search intent, reducing trial-and-error interactions and enabling users to specify service needs (e.g., filtering, aggregation) without requiring implementation details. A preliminary validation conducted in the scope of a real-world case study within the MICS project [8] highlighted the benefits of using metadata-informed prompts in improving discovery efficiency. Future work will focus on devising prompt templates for the LLM-based system designed to suggest additional steps or tasks that could enhance a previously structured scenario, by leveraging specialization relationships between scenarios. Apart from specialization, we will consider the integration of further scenario relationships in the Analysis Scenarios model (e.g., taking inspiration from the renowned UML Use Cases relationships, which permit to specify inclusion and extension between Use Cases), investigating how to employ the LLM-based system to support the users in recognising such relationships. Moreover, from a general perspective, we will introduce personalisation elements in the prompt templates, to tailor interactions and response formats based on users' profiles (e.g., adapting the output writing style depending on the specific skills of domain experts and R&D managers).

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] T. P. Raptis, A. Passarella, M. Conti, Data Management in Industry 4.0: State of the Art and Open Challenges, IEEE Access 7 (2019) 97052–97093.

[2] M. Aiello, I. Georgievski, Service composition in the ChatGPT era, Service Oriented Computing and Applications 17 (2023) 233–238.

[3] J. Zamfirescu-Pereira, R. Y. Wong, B. Hartmann, Q. Yang, Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts, in: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, 2023, pp. 1–21.

[4] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, Retrieval-Augmented Generation for Large Language Models: A Survey, arXiv preprint arXiv:2312.10997 (2023).

[5] Y. Li, B. Starly, Building A Knowledge Graph to Enrich ChatGPT Responses in Manufacturing Service Discovery, Journal of Industrial Information Integration (2024) 100612.

[6] F. Monti, F. Leotta, J. Mangler, M. Mecella, S. Rinderle-Ma, NL2ProcessOps: Towards LLM-Guided Code Generation for Process Execution, in: International Conference on Business Process Management, 2024, pp. 127–143.
[7] Z. Zeng, W. Watson, N. Cho, S. Rahimi, S. Reynolds, T. Balch, M. Veloso, FlowMind: Automatic Workflow Generation with LLMs, in: Proceedings of the Fourth ACM Int. Conf. on AI in Finance, 2023, pp. 73–81.
[8] MICS - Made in Italy Circolare e Sostenibile, 2025. URL: https://www.mics.tech/en/home/, Accessed on March 2025.
[9] D. Bianchini, M. Garda, M. Melchiori, A. Rula, Leveraging Large Language Models for Data Service Discovery, in: 2024 IEEE International Conference on Web Services (ICWS), 2024, pp. 1097–1099.
[10] Q. Huang, Z. Wan, Z. Xing, C. Wang, J. Chen, X. Xu, Q. Lu, Let's Chat to Find the APIs: Connecting Human, LLM and Knowledge Graph through AI Chain, in: 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2023, pp. 471–483.
[11] The OpenAPI Specification, 2024. URL: https://spec.openapis.org/oas/v3.1.0, Accessed on December 2024.
[12] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, D. C. Schmidt, A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT, arXiv preprint arXiv:2302.11382 (2023).
[13] Prompt Chaining – Introduction and Use Cases, 2024. URL: https://www.promptingguide.ai/techniques/prompt_chaining, Accessed on December 2024.
[14] LangChain framework, 2024. URL: https://python.langchain.com/v0.1, Accessed on December 2024.
[15] ChromaDB, 2025. URL: https://www.trychroma.com/, Accessed on March 2025.
[16] *mixedbread-ai/mxbai-embed-large-v1* Embedding Model from the HuggingFace Platform Hub, 2025. URL: https://huggingface.co/mixedbread-ai/mxbai-embed-large-v1, Accessed on March 2025.
[17] R. D. Pesl, J. G. Mathew, M. Mecella, M. Aiello, Advanced System Integration: Analyzing OpenAPI Chunking for Retrieval-Augmented Generation, arXiv preprint arXiv:2411.19804 (2024).
[18] LangChain - Multi-Vector Retriever Interface, 2025. URL: https://python.langchain.com/docs/how_to/multi_vector/, Accessed on March 2025.
[19] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, et al., Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena, Advances in Neural Information Processing Systems 36 (2024).
[20] Ragas – A framework for the evaluation of Retrieval Augmented Generation (RAG) pipelines, 2024. URL: https://docs.ragas.io/en/stable/index.html, Accessed on December 2024.
[21] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, A. Chadha, A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications, arXiv:2402.07927 (2024).