SN Computer Science (2026) 7:339
https://doi.org/10.1007/s42979-026-04863-3

ORIGINAL RESEARCH

Deploying Conversational Agents in Virtual Research Environments: Approaches and Lessons Learned

Massimiliano Assante[1] · **Leonardo Candela**[1] · **Andrea Dell'Amico**[1] · **Luca Frosini**[1] · **Francesco Mangiacrapa**[1] · **Alfredo Oliviero**[1] · **Pasquale Pagano**[1] · **Giancarlo Panichi**[1] · **Biagio Peccerillo**[1] · **Tommaso Piccioli**[1] · **Marco Procaccini**[2]

Received: 22 September 2025 / Accepted: 25 February 2026

Abstract
Conversational agents have the potential to streamline tasks, provide support, and enhance user experience across various domains including Virtual Research Environments (VREs). The recent progress in conversational artificial intelligence and Large Language Models (LLMs) offers novel strategies for the development of these agents. This paper reports on the potential benefits, the challenges and the approaches resulting from concrete experiences in developing and equipping D4Science-based VREs with suitable conversational agents. The paper presents three successive implementation approaches and the resulting agent solution, each designed to address the limitations identified in the preceding iteration and to leverage the advantages offered by newer implementation and development options. The proposed approaches led to the progressive refinement of the agent design and functionality, resulting in DAVE, a conversational agent capable of securely interacting with multiple D4Science services and supporting a wide range of user workflows. The iterative process highlighted critical requirements—including authentication handling, usability, and extensibility—that can inform the design of conversational agents in similar research infrastructures. The study shows that conversational agents can effectively lower the barrier to accessing VRE functionalities and enhance user engagement. The resulting design principles and lessons learned provide a foundation for future work aimed at extending DAVE with an enhanced feedback mechanism and locally hosted LLM integration, and conducting systematic usability evaluations within active research communities.

Keywords Virtual Research Environment · Research infrastructure · Conversational agent · AI Agent · Multi-agent systems

Massimiliano Assante, Leonardo Candela, Andrea Dell'Amico, LucaFrosini, Francesco Mangiacrapa, Alfredo Oliviero have contributed equally to this work.

✉ Leonardo Candela
leonardo.candela@isti.cnr.it

Massimiliano Assante
massimiliano.assante@isti.cnr.it

Andrea Dell'Amico
andrea.dell'amico@isti.cnr.it

Luca Frosini
luca.frosini@isti.cnr.it

Francesco Mangiacrapa
francesco.mangiacrapa@isti.cnr.it

Alfredo Oliviero
alfredo.oliviero@isti.cnr.it

Pasquale Pagano
pasquale.pagano@isti.cnr.it

Giancarlo Panichi
giancarlo.panichi@isti.cnr.it

Biagio Peccerillo
biagio.peccerillo@isti.cnr.it

Tommaso Piccioli
tommaso.piccioli@isti.cnr.it

Marco Procaccini
marco.procaccini@igg.cnr.it

1 Institute of Information Science and Technologies “Alessandro Faedo”, National Research Council of Italy, via G. Moruzzi, 1, 56124 Pisa, Italy

2 Institute of Geosciences and Earth Resources, National Research Council of Italy, via G. Moruzzi, 1, 56124 Pisa, Italy

Published online: 04 April 2026

Introduction

The recent appearance of Large Language Models (LLMs) in the landscape of Artificial Intelligence (AI) fueled an unprecedented flourishing of services pertaining to Natural Language Processing (NLP). Among these, AI agents have emerged as one of the most transformative applications [1–3]. Powered by the advanced language understanding and generation capabilities of LLMs, they can engage in coherent, context-aware dialogues. Their ability to interface with a variety of services, invoke external API calls, and collect results makes them a versatile tool across a wide range of domains, such as customer service, entertainment, healthcare, and scientific research.

In the context of scientific research, Virtual Research Environments (VREs) [4] and Science Gateways [5] are increasingly being adopted as platforms to support open, collaborative, data-intensive, and possibly computationally demanding research work within diverse communities across the boundaries of research performing organizations. They aim to lower the barrier to performing research, making it more reproducible, collaborative, and scalable by hiding technical complexity, democratizing access to data and computing resources, promoting the sharing of research products by the "as early as convenient" approach, and giving researchers intuitive tools for their domain.

AI agents have a strong potential to improve the usability and accessibility of these research-oriented platforms as well as their developments [6, 7]. They might bring benefits from many points of view, including real-time support through chatbots, advanced search capabilities based on semantics, virtual assistance in research tasks, gateway improvement driven by sentiment analysis, automated gateway scanning for accessibility compliance, identification of compromised accounts and security threats.

This paper reports on our experiences and lessons learned while developing conversational agents within the D4Science infrastructure and its VREs [8–10]. Specifically, we describe three subsequent efforts in designing and implementing a conversational agent suitable for the D4Science settings: namely (*a*) the early prototype *Janet* [11], i.e., a modular agent based on a pipeline of fine-tuned components; (*b*) our second attempt *D4Science AI Agent* [12], i.e., a multi-tool agent based on the Cheshire Cat [13, 14] framework; and (*c*) the current proposal *DAVE* (D4Science Assistant for Virtual research Environments), i.e., a highly modular multi-tool multi-agent system based on the Google Agent Development Kit (ADK) framework [15]. For each system, we discuss the design rationale, identify the limitations and shortcomings that emerged in practice—our "lessons learned", and explain how these guided the development of the subsequent iteration. Through this analysis, we aim to elucidate the challenges associated with integrating conversational agents into VREs and propose an approach that may serve as a reference for similar initiatives.

Throughout this work, we have been guided by a set of requirements that we argue are broadly applicable to any conversational agent designed for D4Science-based VREs—and, more generally, to agents deployed within VREs or Science Gateways. These requirements were:

Assistance Assist researchers at least in question answering, resource retrieval, paper summarization, and integration with the underlying services;

Flexibility Enable modular replacement to accommodate technological advances;

Security Guarantee at least the same level of safety and security provided by the underlying infrastructure;

Extensibility Be extensible to meet the evolution and growth of the underlying infrastructure concerning capabilities and capacity;

Feedback Evolve in response to user interactions.

The main contributions of this paper are summarized below:

- We share our first-hand experience in designing and deploying a conversational agent integrated in a large-scale research infrastructure, highlighting both the challenges encountered and the lessons learned that may inform future initiatives;
- We introduce DAVE, a novel conversational agent tightly integrated in the D4Science VREs, illustrating how its design principles can address practical challenges, offering insights for researchers developing similar systems;
- We present DAVE's practical value through representative use cases, showing how the tight integration of DAVE into D4Science VREs can enhance researchers' daily workflows.

The remainder of the paper is organized as follows. Section "Background and Related Works" reviews relevant background and related work. Section "D4Science AI Agents" describes the three conversational agents introduced above. Section "Use Cases" demonstrates how the current D4Science agent DAVE performs in representative use cases. Finally, Section "Conclusion and Future Work" concludes the paper and outlines directions for future work.

Background and Related Works

Developing a conversational agent for a Virtual Research Environment (VRE) differs significantly from developing one for a traditional environment [16], e.g., the distinctive characteristics of VREs, in terms of content and capabilities, along with their evolving nature, present specific challenges. This section provides an overview of the VRE concept and its implementation within the D4Science infrastructure, examines Large Language Models and conversational agents, reviews relevant technologies and frameworks that can support agent development, and summarizes prior experiences in developing such agents.

Virtual Research Environments and the D4Science Infrastructure

Virtual Research Environments (VREs), also referred to as Science Gateways or Virtual Laboratories, are web-based platforms designed to support the needs of specific research communities [4, 5]. They are conceived to provide the tools and resources necessary for achieving particular research objectives, offer their services in an open and flexible manner, and facilitate controlled, fine-grained sharing of research outputs, with an emphasis on ownership, provenance, and attribution.

D4Science [8–10, 17] is a distributed infrastructure powered by the gCube technology [18] to provide integrated VREs *as-a-service* to research communities. D4Science was designed to conceptually play the role of a central-hub offering seamless access to its own resources (datasets, services, computing and storage capacity) as well as to services and computing capacity offered by other infrastructures and service providers. From the end-user perspective, D4Science-based VREs are web-based working environments that include various services related to four main components: a shared workspace, a social networking area, a data analytics platform, and a publishing platform [8]. Their level of maturity and deep integration promote a workflow between researchers based on cooperation and sharing. Researchers can share their material and access materials shared by others in the workspace, where each item is equipped with a unique identifier that allows unambiguous reference to it from the other components. Social posts can be used to communicate and discuss opinions among VRE members. The data analytics platform allows defining, publishing, and invoking methods and workflows, promoting their reuse as well as the reproducibility of results obtained by them [19]. Finally, the publishing platform permits announcing and being informed about the availability of research artifacts at different maturity levels.

At the moment of writing, the D4Science Infrastructure serves more than 27,000 users in 90+ countries all over the world by 220+ active VREs [17]. D4Science VREs have been adopted by dozens of research communities in about 20 years [10]. A list of currently active gateways can be found here [20].

Large Language Models and Conversational Agents

LLM [21, 22] is the term generally used to denote a neural language model based on the Transformer network architecture [23], which usually contains billions and even trillions of parameters. LLMs are pre-trained with enormous amounts of textual data, from which they learn grammar rules and acquire vast world knowledge. They exhibit strong language understanding and generation capability, and their abilities go even beyond basic natural language processing tasks, including: in-context learning, instruction following, and multi-step reasoning [21]. Currently, LLMs are conceived as a fundamental building block for general-purpose AI agents, to the point that they are the most promising technology towards so-called Artificial General Intelligence (AGI) [1].

There are many LLM examples, with the most notable ones pertaining to OpenAI's GPT family, Google's Gemini family, Meta's Llama family, and Anthropic's Claude family. To name just a few examples, the GPT family includes GPT-4 [24] and GPT-5 [25], at the core of ChatGPT [26], the famous chatbot released in November 2022 which is arguably the first LLM-powered service to achieve widespread adoption among the general public. Google's Gemini family evolved from PaLM [27] and forms the backbone of Google's homonymous chatbot [28]. The Llama family includes Llama-3 [29] and Code Llama [30], two *open-weight* models, with the latter being a code-oriented LLM which has been trained with extensive code-specific datasets to acquire advanced coding capabilities. Finally, the models of the Claude family (e.g., Claude 3 [31]) are trained using the so-called Constitutional AI [32], which is based on giving *principles* to guide the learning process. In addition to general-purpose LLMs, specialized Scientific Large Language Models have been devised to better handle domain-specific terminology and facilitate the understanding of scientific language, e.g., [33].

Conversational agents, or *chatbots*, [3, 34] are AI models designed to engage in meaningful and interactive conversations with users. For them, the ability to simulate human-like dialogue abilities is paramount, as it allows users to interact with them in a fluid and natural manner. Before LLMs, they were limited mainly to keyword-based and pattern-based communications in which user questions were matched with a set of answers in a database.

The advent of LLMs completely revolutionized this landscape. LLMs allowed conversational agents to evolve to the point that they are becoming an integral part of an ever-increasing amount of services and applications, such as virtual assistants (e.g., Siri and Alexa), customer support in businesses, personal assistance in healthcare, personal tutoring in education, player guidance in digital gaming [3]. They are helping users improve their abilities and companies provide complex services with limited operational costs, and this trend is not going to stop in the foreseeable future.

Conversational Agents Enabling Technologies

Several technologies and frameworks are emerging to support the development of conversational agents, e.g., LangChain [35], Qdrant [36, 37], Cheshire Cat [13, 14], and the Google Agent Development Kit (ADK) [15] have supported our experiences. In addition, to facilitate the integration of agents in specific contexts, some *mediators* might be needed. Below, we describe the D4Science Python Library, i.e., a software component that facilitates the interaction between an agent and D4Science services.

LangChain [35] is an open-source framework that supports the development of applications powered by LLMs by providing modular components for prompt management, memory handling, and external tool integration. It enables the construction of complex, context-aware workflows where language models interact with data sources, APIs, and computational tools.

Qdrant [36, 37] is a high-performance vector search engine and database, specifically designed to manage and query large-scale collections of high-dimensional vector embeddings. It is commonly employed as the persistence and retrieval backbone in systems requiring semantic search capabilities, most notably in Retrieval-Augmented Generation (RAG) architectures [38]. In such systems, unstructured data, such as text documents, is first segmented into manageable chunks. Each chunk is then processed by an embedding model to produce a dense vector representation, which is subsequently indexed in a Qdrant collection along with its original content and structured metadata stored as a JSON payload.

The core retrieval mechanism involves converting an incoming query into a vector and executing an Approximate Nearest Neighbor (ANN) search within Qdrant to find the indexed vectors with the highest semantic similarity. Implemented in Rust and accessible via gRPC and RESTful APIs, Qdrant is designed for low-latency performance and scalability, making it a frequent choice for production-grade AI applications [39–41].

Cheshire Cat [13, 14] is an open-source framework for building LLM-powered AI agents. It is designed to be highly extensible and language-model agnostic, as it supports integration with a wide range of LLMs via API keys. Its architecture is API-first, enabling seamless interaction through HTTP and WebSocket endpoints. It is fully *dockerized* for easy deployment.

The framework supports RAG functionalities to ground responses in a persistent knowledge base, implemented by the means of a Qdrant vector database called “Rabbit Hole”. This stands at the core of an *episodic memory*, which tracks past interactions, and a *declarative memory*, which stores user-provided documents. Cheshire Cat can be customized and extended through Python plugins, with a rich ecosystem of over 100 plugins already available. Its persistent long-term memory ensures that agents retain context across restarts, and its document ingestion capabilities allow for advanced, context-aware AI behaviors. The active developer community and comprehensive documentation further support rapid development and deployment of intelligent agents.

The framework leverages so-called *tools* to act as bridges between the language model and external functionalities, such as APIs, databases, or custom logic, enabling the agent to perform actions beyond pure text generation. Each tool is developed as a Python library and can be easily registered and integrated into the agent’s workflow.

Google Agent Development Kit (ADK) [15] is an open-source framework for the development of production-ready AI agents on top of LLMs. It provides a structured environment that facilitates the development of both single-agent systems and complex multi-agent architectures. The ADK’s core strength lies in its modular design, which promotes a clear separation of concerns by distinguishing between an agent’s core logic, its tools, and its instructional prompts. This architectural approach enhances maintainability and scalability, allowing developers to build robust systems where specialized agents handle discrete tasks. For instance, a common pattern involves a root agent acting as a router, delegating tasks to sub-agents with specific capabilities, such as data retrieval or code execution.

The framework is built on an asynchronous model, enabling efficient I/O operations and parallel tool execution, which is critical for performance in real-world applications. State management is a central feature that provides a mechanism for agents to maintain context and pass data throughout a session’s lifecycle. For tasks requiring human oversight, the ADK supports Human-in-the-Loop patterns, allowing for crucial approval steps in sensitive workflows. Furthermore, the inclusion of a comprehensive evaluation suite allows for the systematic testing of agent performance, ensuring reliability and preventing regressions.

A desirable feature of Google ADK is that agents implemented with it are compliant with the Model Protocol for Communication (MPC) and the Agent-to-Agent (A2A) protocols [42], ensuring interoperability in multi-agent ecosystems. This compliance further strengthens the ADK's suitability for distributed environments, where heterogeneous agents need to exchange information and collaborate reliably.

The *D4Science Python Library* is a comprehensive software component designed to facilitate seamless and unified access to the diverse suite of RESTful APIs offered by the D4Science infrastructure. Developed with extensibility and usability as primary objectives, the library exposes a command-line interface (CLI) that abstracts the complexity of interacting with multiple D4Science services, including the Workspace, Social, Catalogue, and Cloud Computing Platform (CCP). By offering a single entry point for API consumption, it lowers the barrier for both end-users and developers who need to automate, script, or integrate D4Science functionalities into their workflows.

The library is organized around modular client classes, each corresponding to a specific D4Science service. These clients encapsulate HTTP communication, authentication (including OAuth2 flows), error handling, and data serialization, thereby providing a robust and consistent interface for higher-level operations. The CLI, implemented using the Typer framework [43], builds on these clients to offer a user-friendly and discoverable command structure. Through this interface, users can perform a range of operations—such as file uploads and downloads, folder management, metadata manipulation, and social interactions—via intuitive commands.

The design of the D4Science Python Library was driven by the primary objectives of extensibility, usability, and production readiness, allowing it to abstract the complexity of interacting with multiple D4Science services. Its configuration system supports multiple deployment environments (e.g., development and production) and provides secure credential management. The library implements robust error-handling mechanisms that offer clear feedback for both expected and exceptional conditions. Furthermore, it is designed for seamless integration into automated agents and plugins, exposing both command-line and programmatic interfaces to facilitate adoption across a wide range of scientific and research applications.

Conversational Agents in Research

AI agents have recently attracted significant attention, resulting in a growing body of work that explores their architectures, applications, and design principles. The interest has been so pronounced that various surveys have already been published on the topic [1, 2, 44, 45].

When focusing on scientific practice, there are also many examples in which AI agents have been employed to support researchers in diverse activities.

For instance, a paper by Yu et al. [46] systematically overviews "Artificial Intelligence for Research" (AI4R) applications. The study surveys existing literature, identifying prominent AI tools that support different stages of the research process, ranging from literature review and hypothesis generation to experimentation, evaluation, and dissemination. It also highlights examples of AI-enabled scientific advances as evidence of AI's increasing integration in research workflows. Beyond individual tools, the authors emphasize the role of AI in shaping research ecosystems and enabling new models of human–AI collaboration.

Beyond this general analysis, the literature also reports numerous studies in which particular aspects of AI-aided research are addressed.

In [47] the authors propose a conceptual framework to address reliability challenges in AI-driven scientific workflows. The work argues that foundation model-powered agents are prone to generating misleading or hallucinated outputs that can corrupt downstream results. The presented approach integrates agent-level safeguards and workflow-level instrumentation to enhance reliability and reproducibility in AI-assisted scientific discovery. Their framework allows for early detection, containment, and recovery from erroneous behavior, thereby improving the rigor and trustworthiness of agentic systems.

Google researchers present an AI co-scientist [48], a multi-agent system built on Gemini 2.0 designed to generate and refine scientific hypotheses through a "generate–debate–evolve" approach. The system features asynchronous task execution for scalable computation and a tournament process for self-improving hypotheses. While general in scope, its validation focuses on biomedical domains, including drug repurposing, target discovery, and bacterial evolution. Reported results demonstrate experimentally validated findings, suggesting the potential of multi-agent architectures to augment scientific discovery.

There are also works that analyze AI-aided research in specific fields.

One of the first examples is Bert [49]. It is a lightweight, context-linked event notification and virtual assistant system used by a distributed team of astrophysicists. Integrated into their chat client, Bert's primary functions are to announce relevant events, like approaching sunrise, and to respond to user queries for information from a shared database. It was designed to improve team communication, awareness, and performance during time-critical operations, reducing human error.

Another example is InfoBot [50], a chatbot for a university course, that serves as a model for a virtual assistant in a research infrastructure. By instantly answering student questions on course materials and logistics from a shared knowledge base, the system demonstrates how such a tool can automate information retrieval for collaborative, data-rich projects.

A similar purpose is achieved by Yang and Niu [51]. They propose a multi-agent framework to create a virtual teaching and research intelligence system for lecturers. The system's agents provide personalized student guidance, assist with research tasks like literature screening, and manage campus resource allocation. The framework is designed to enhance productivity and optimize the use of educational and scientific resources within a research infrastructure.

Gao et al. [52] discuss the role of AI agents in biomedical discovery, envisioning compound systems that combine LLMs, machine learning models, and experimental platforms. They classify agents by autonomy, from assistants handling narrow tasks to systems capable of hypothesis generation and experimental design. Applications include virtual cell simulation, programmable phenotype control, and cellular circuit design. The authors also note open challenges concerning reliability, ethics, and responsible deployment, framing AI agents as potential contributors to future scientific practice.

Again in the biology domain, Harrer et al. [53] argue that generative AI agents are reshaping biology by enabling high-resolution functional genome annotation and the integration of multimodal biological data. They outline how LLM-based agents can autonomously plan and execute workflows, contextualize complex datasets, and design experimental pipelines, thereby advancing synthetic and systems biology. The paper highlights applications in protein structure prediction, drug discovery, and the development of digital cells, while also stressing the importance of high-quality datasets, ethical standards, and responsible deployment.

In Software Engineering, He et al. [54] analyze how multi-agent systems based on LLMs are applied in this discipline, surveying recent studies across the software development lifecycle. They show, through case studies, how multiple specialized agents can collaborate to address complex tasks, highlighting both current capabilities and limitations. The paper also identifies open research gaps and proposes a research agenda aimed at improving agent autonomy, synergy, and scalability in real-world software projects.

Zhang et al. [55] describe a framework called GeoGPT supporting geospatial data collection, processing, and analysis in an autonomous manner. This framework relies on an LLM to understand the demands of users, and then think, plan, and execute defined GIS tools sequentially to output final effective results. It is conceived as an assistant for GIS professionals accepting natural language instructions as input and adapting to various geospatial tasks, which can serve as an assistant for GIS professionals.

Finally, in the VRE domain, Sperhac et al. [7] highlight opportunities for integrating AI into research computing infrastructures. Through the NSF SGX3 initiative, they initiated a Blueprint Factory process to map future technical requirements. They identified key themes—such as interactive user assistance, usability, computation and training, and data handling—emphasizing both opportunities and challenges. The study positions AI-driven gateways as a

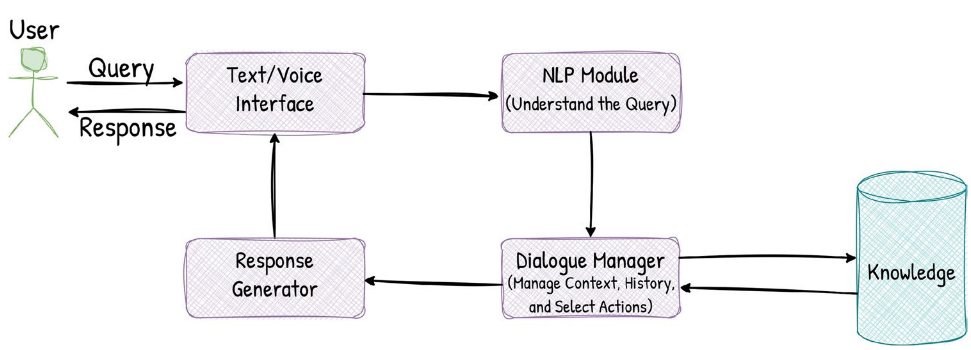

Fig. 1 Janet general architecture, adapted from [11]

promising direction for democratizing access to advanced computing resources.

D4Science AI Agents

In this section, we present the three conversational agents we developed, i.e., Janet (cf. Section "Janet", early 2023), D4Science AI Agent (cf. Section "D4Science AI Agent", late 2024), and DAVE (cf. Section "DAVE", current).

Janet

Janet was an LLM-powered conversational agent that aimed to take advantage of the knowledge-rich D4Science VREs to help researchers through an interactive conversational interface [11].

Janet was designed with a modular approach. Figure 1 highlights its main modules:

Text/Voice Interface: allows user interaction using natural language—either by typing or speaking;

NLP Module: understands user queries to decide what functionality the user is looking for;

Dialog Manager: decides the right tool based on the NLP Module input and the dialog history, possibly accessing a knowledge source to generate the response;

Response Generator: builds a human-like reply based on the input from the Dialog Manager.

The NLP Module includes an Intent Classifier based on a distilled version of BERT [56], which was trained with 275 manually-labeled sentences; an Entity Extractor based on RoBERTa [57], trained on 97 tuples, each containing a sentence and its entities; and an Offensive Language Classifier trained to flag potentially offensive queries.

The Response Generator includes a Neural Retriever, a Language Generator, and a Recommender. The Neural Retriever is an MPNet-base [58] sentence transformer that maps paragraphs into dense 768-dimensional vectors. It was trained with 313,156 examples in the form of <query, context> pairs extracted from the user manual of D4Science, the MS-Marco v2.1 dataset [59], the PubMed QA dataset [60], the QAsper dataset [61], and the ScienceQA dataset [62]. The Language Generator is a fine-tuned T5 LLM [63], chosen over more powerful alternatives (e.g., at the time, GPT−3.5 [64] and GPT-4 [24]) due to its availability for free. It was trained with a dataset containing examples in the form <text, target-text>, with "text" being the query augmented with task-specific prefixes for each task we wanted to support. Recommender is implemented by profiling the user's interests, which are extracted from their queries. To do this, an entity extractor is used to identify topics or resources of interest from user natural language queries. Then, these interests are ranked according to their frequency and recency.

These tools are augmented with a knowledge base that is constructed from the resources available in the VRE. An ideal knowledge base would encompass various forms of knowledge representations, which may include, but are not limited to, a graph database and a vector database. However, in Janet, we used a vector index for simplicity, implemented using FAISS [65].

Lessons Learned

Although Janet represents a valuable effort to integrate conversational agents within D4Science VREs, we encountered some limitations on the Assistance, Flexibility, Extensibility, and Feedback requirements. We could still work on Janet to improve it on the Assistance side, but its limitations on the Flexibility, Extensibility, and Feedback sides brought us to explore different design choices. In particular:

Flexibility—Despite Janet's modular design, its reliance on task-specific trained components (e.g., Intent Classifier, Entity Extractor, Language Generator) can limit flexibility when adapting or upgrading individual modules. In particular, integrating newer language models or retrieval strategies may require retraining several components to preserve system coherence.

Moreover, Janet's architecture relies on a rigid, multi-stage pipeline where user input flows through a sequence of dedicated NLP modules (e.g., Intent Classifier, Entity Extractor) before a response is generated. This predefined execution path is inherently static and limits the agent to a few workflows, determined by matching the user's intent over a predetermined set.

Extensibility—Extending Janet's capabilities requires significant engineering effort. The introduction of a new function requires a cycle of data collection, manual labeling, and the subsequent retraining of multiple fine-tuned models. This process may be time-consuming and demands

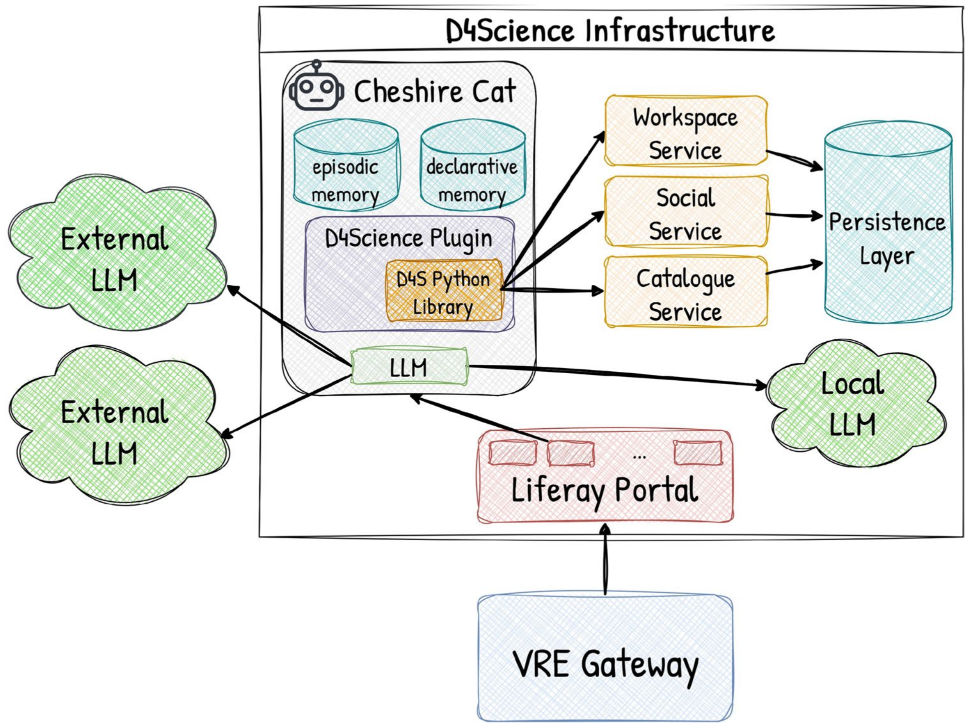

Fig. 2 D4Science AI Agent general architecture, adapted from [12]. It is organized as a multi-tool single-agent with a flexible LLM binding and two knowledge bases, which are managed as a short-term and a long-term memory, respectively

specialized expertise in model training. This approach, while state-of-the-art at the time of its conception, presents a considerable barrier to rapid development and iteration.

Feedback—Although it could answer queries, generate recommendations, and extract entities, it lacked mechanisms to dynamically incorporate user feedback into its models or knowledge representations. The vector-based knowledge index provided a static view of resources, and the Recommender relied on simple frequency and recency metrics rather than adaptive learning.

D4Science AI Agent

In the months following the completion of Janet, an ever-growing interest in AI agents created a fertile ground for the emergence of novel tools aimed at agent development. We found Cheshire Cat [14], described in Conversational Agents Enabling Technologies, particularly interesting due to its plugin-based design, active community, and user-friendliness. Consequently, we restructured our prototype into a more modern and efficient architecture by adopting Cheshire Cat as the primary development framework [12].

The architecture of the resulting prototype—the D4Science AI Agent—is shown in Fig. 2.

The first major discontinuity with respect to Janet was the inclusion of an LLM in the most flexible way possible. Instead of relying on a finely-tuned one, we leveraged the features of Cheshire Cat to enable LLM selection via API key, thus supporting both external LLMs and self-hosted ones. This flexibility paves the way for the elimination of a significant external dependency altogether, thereby ensuring that user data remain within the data-center and allowing for their handling with greater confidentiality.

Since LLM selection was deliberately left as a degree of freedom in our design, a mechanism was required to provide the appropriate operational context, to guide the behavior of the D4Science AI Agent and ensure consistency in its interactions. We designed a structured “prompt prefix” inspired by best practices in prompt engineering [66]. This *preamble* serves as a persistent instruction set that is injected at the beginning of every interaction session with the underlying language model. It characterizes the agent’s role, capabilities, and boundaries, and embeds a lightweight personality aligned with the D4Science mission. Specifically, the prefix defines the agent as a helpful, knowledgeable, and efficient assistant capable of executing tool-based actions and providing high-level summaries of complex resources.

The main user interface, implemented as a web page that provides access to the chatbot, is hosted by the VRE Gateway, which acts as the web-based entry point to the D4Science VREs. All communication with the D4Science services (i.e., Workspace, Social, and Catalogue in Fig. 2) is provided by the D4Science Plugin, which implements the tools based on the D4Science Python Library (presented in Conversational Agents Enabling Technologies). These tools are implemented as Python self-describing functions, each documented at their entry point. Upon receiving a user request, the framework leverages the LLM to interpret the user's intent and determine whether the requested action can be fulfilled by any of the available tools. If so, the LLM is responsible for extracting and mapping the necessary parameters from the user's input according to the tool's specification, ensuring that the tool can be correctly invoked by the framework. The selected tool is then executed, and the resulting output is processed and presented to the user as part of the agent's response.

During user-agent interaction, Cheshire Cat's RAG functionalities may be triggered. These include four possible actions: indexing/retrieving user-agent interactions into/from episodic memory, or indexing/retrieving documents into/from declarative memory. Both episodic and declarative memory are Qdrant-based vector databases that store embeddings, obtained from original information through chunking and applying an embedding mode. Both memories expose an interface to interact with them programmatically, for instance, in the implementation of a tool.

Lessons Learned

D4Science AI Agent represented a step forward with respect to Janet, addressing mainly its flexibility and extensibility shortcomings. However, it still presented some issues. We did not include a feedback mechanism, but we could obtain it with a reasonable effort by leveraging the functionalities provided by Cheshire Cat. For instance, by intercepting user interactions with the agent and processing them to feed episodic and/or declarative memories. Also on the security side, Cheshire Cat does not include native multi-user support, which is necessary in the context of D4Science VREs. Also in this case, we could obtain it with a reasonable effort.

What convinced us to look for a novel solution and thus restructure the whole architecture was again on the flexibility and extensibility sides. In particular:

Flexibility, Extensibility—The adoption of Cheshire Cat as the foundation for the D4Science AI Agent provided significant advantages in terms of modularity. However, it also limited the architecture to a single-agent, multi-tool model. While effective for simple deployments, this approach proved less suitable in the context of VREs, where multiple specialized agents might naturally co-exist—possibly being developed by different researchers. Furthermore, since agent

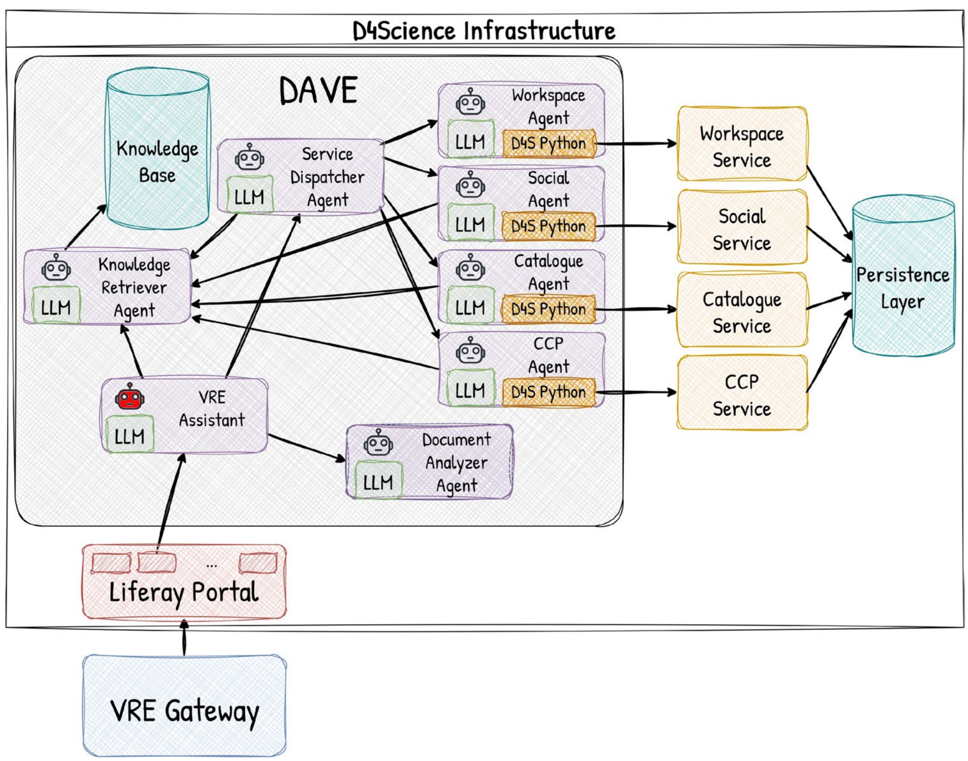

Fig. 3 DAVE general architecture. It is designed as a system of specialized agents with a VRE Assistant acting as a central orchestrator. Some agents interface towards D4Science services through the D4Science Python Library. Each agent in the system has its own LLM binding which is independent of the others

behavior customization relied heavily on prompt engineering, concentrating all functionalities within a single agent required maintaining a very large and complex prompt, increasing operational costs and diluted the relevance of contextual information during interactions. The rapid evolution of the D4Science VRE ecosystem accentuated these issues, convincing us to migrate to a novel solution.

DAVE

D4Science Assistant for Virtual research Environments, or simply DAVE, is our final proposal of a conversational agent in the D4Science environment. It is designed to be as flexible as possible, to limit the need for re-design and re-engineering while facilitating the seamless integration of future advancements with minimal or no required modifications.

Figure 3 shows its architecture.

Similarly to the D4Science AI Agent, users gain access to DAVE through the VRE Gateway, the web-based entry point to the D4Science ecosystem. Rather than being limited to a generic chat with the assistant, the Gateway presents specific web interfaces depending on the context in which it is accessed, allowing DAVE to be seamlessly integrated into service-specific pages within the VREs. In its simplest form, DAVE is available through a chatbot-like interface, enabling direct conversational interactions. This flexible system for delivering dynamic content is possible because the Gateway is served by the Liferay Portal, a modular Java-based web platform that delivers dynamic content and applications while handling client-side HTTP requests.

The Gateway is expected to give seamless access to services and resources hosted on D4Science premises or federated to it. Among the services, there are core ones encompassing all the Virtual Research Environments and communities:

Workspace: The service providing users with data hosting, sharing, and retrieval à-la remote file system;

Social: The service providing users with social-networking functionalities such as content posting and user interaction;

Catalogue: The service allowing users to publish and retrieve research artifacts;

CCP: The service providing users with data analytics facilities where users can define, publish, and run algorithms using D4Science computational resources.

Agents

DAVE is composed of a central, autonomous orchestrator agent and a set of specialized sub-agents. The key agents are:

VRE Assistant: The central and autonomous orchestrator agent. It is responsible for receiving user requests, determining the correct course of action, and delegating tasks to its specialized sub-agents. It is also responsible for synthesizing the information and preparing the final response to be sent back to the user.

Knowledge Retriever Agent: A specialized sub-agent that serves the VRE Assistant. It manages the entire Retrieval-Augmented Generation (RAG) system. Its role is to retrieve relevant information from the Knowledge Base (which is part of the RAG system) and to process user-provided content for inclusion. This information is then used to enrich prompts for other agents.

Service Dispatcher Agent: A specialized sub-agent that acts as a routing point. It translates user requests into specific D4Science service operations and forwards them to the correct service agent. It may be called by the VRE Assistant to retrieve contextual information or execute a specific service request.

Document Analyzer Agent: A specialized sub-agent for processing various types of textual documents, including academic papers. Its primary role is to perform a specialized analysis based on the document type, extracting and defining metadata (e.g., title, authors, geographical references, and document type), identifying relevant tags, and producing concise summaries. The agent processes the document to make its content available for other agents to use.

Social Agent: Manages interactions with the Social service, processing and summarizing user activities like posts and comments;

Workspace Agent: Handles tasks related to the Workspace service, such as searching and organizing documents;

Catalogue Agent: Facilitates the querying of the Catalogue for searching and retrieving resources.

CCP Agent: Manages interactions with the Common Collaboration Platform (CCP) for user and workspace management.

A typical user interaction begins when the **VRE Assistant** receives a request, which interprets the input and plans the sequence of actions required to fulfill it. These actions may involve delegating specific tasks to specialized sub-agents, processing their outputs, and possibly forwarding intermediate results to other agents for further refinement. Throughout this process, the underlying LLM serves as the central reasoning component, dynamically evaluating the context and adapting the plan as needed. Each agent is further equipped with a set of tools, implemented in Python, that provide concrete functionalities to support task execution.

LLM selection in DAVE follows the approach introduced in our previous prototype, where models can be configured via API key, supporting both external and, in the future, locally hosted instances. In addition, DAVE's multi-agent architecture allows each agent to maintain its own LLM binding independently of the others. This enables the use of different models according to task requirements, employing larger and more advanced models for complex tasks while relying on lighter ones for simpler tasks. For instance, in the current configuration, **VRE Assistant** and **Document Analyzer Agent** use Gemini 2.5 Pro, while the agents interfacing towards the services leverage Gemini 2.5 Flash-Lite.

Knowledge Base

The Knowledge Base is implemented as a vector database using Qdrant. It stores embeddings from various resources and supports efficient similarity search, enabling the retrieval of semantically relevant information to ground the responses. Information is organized into three main categories. *General information* includes factual knowledge about D4Science, its open science mission, the concept of Virtual Research Environments, and other background elements useful to define the context. *VRE-specific information* captures knowledge related to the particular VRE in which the agent is deployed, such as domain-specific content in marine biology, geology, sociology, etc. Finally, *user-specific information* is collected during user interactions or derived from the analysis of user-provided documents, thereby personalizing the agent's responses to individual research needs.

The embeddings stored in the vector database are tagged to indicate their information category. This tagging plays a crucial role in ensuring that retrieval is both relevant and context-aware. General information is available system-wide and is retrieved whenever it is semantically relevant to the user's request. VRE-specific information, by contrast, is retrieved only within the scope of the VRE in which the agent is deployed, thus preventing cross-domain interference (e.g., marine biology content surfacing in a geology environment). Finally, user-specific information is accessible solely in the context of the user who produced it, either through interactions or document analysis, thus ensuring personalized responses without compromising data separation.

Feedback

To continuously improve system performance and reliability, DAVE incorporates a feedback mechanism that collects and processes anonymized interactions between users and agents. Each interaction—including user queries, agent responses, and tool invocations—is logged in an anonymized manner to ensure no personally identifiable information is retained. Metadata such as interaction type, response latency, and success/failure indicators are preserved to enable systematic analysis. We currently aggregate key quantitative metrics, such as the number of tool calls, retrieval success rates, and average response length. For future work, we plan to implement a mechanism for qualitative evaluation, such as analyzing user-provided ratings or inferred satisfaction signals.

Lessons Learned

DAVE's architecture, compared to our previous prototype, better embodies the requirements expressed in the Introduction. In particular:

- **Assistance**—DAVE empowers researchers by offering capabilities for question answering, resource retrieval, and document summarization. It is seamlessly integrated with the main D4Science services—Workspace, Social, Catalogue, and CCP—through dedicated APIs, allowing for direct interaction and utilization of the platform's core functionalities.
- **Flexibility**—DAVE is architected as a modular multi-agent system, featuring a central orchestrator agent and specialized sub-agents. These sub-agents are defined by their instruction prompts and a specific set of tools. The architecture ensures a loose coupling between agents and their LLMs, as it supports the integration of diverse LLMs from various providers or self-hosted models. Each sub-agent can be configured to use the most suitable LLM for its specific tasks, allowing for the easy substitution of models without requiring significant architectural modifications.
- **Security**—Users interact with DAVE within the D4Science VRE, thereby leveraging its robust authentication mechanisms. All information exchanged with DAVE and stored in the Knowledge Base is categorized into general, VRE-specific, and user-specific data. This design guarantees data privacy and prevents cross-user data access by ensuring that user-specific information is isolated and accessible only to the respective user.
- **Extensibility**—The DAVE architecture is designed for extensibility. *Novel micro-functionalities* can be integrated by implementing specific tools within existing sub-agents. For broader, *macro-level functionalities*, new, ad-hoc sub-agents can be seamlessly added to the system. The infrastructure is inherently ready for

Fig. 4 Example user—DAVE interaction: document retrieval and analysis. The complete transcript is in Appendix A

User: Can you list the content of my `papers` folder?

DAVE: Here are the contents stored in your folder:

- `Enacting_open_science.pdf`
- `Open_Science_Survey.pdf`

User: Please analyze `Enacting_open_science.pdf`.

DAVE: Here is a summary of the document `Enacting_open_science.pdf`:
Summary: This paper describes the D4Science platform, a technological infrastructure designed to support and enact the principles of Open Science. ...
The core components of the D4Science offering include:

1. A shared workspace for storing, organizing, and versioning research artefacts.
2. A social networking platform for communication, discussion, and collaboration among researchers.
3. ...

By integrating these tools, D4Science aims to create a seamless environment where research activities are transparently captured, making the entire process more fair, solid, and democratic.
Authors:

- M. Assante
- L. Candela
- ...

Keywords:

- Open Science
- gCube
- ...

Multi-Party Computation (MPC) and Agent-to-Agent (A2A) communication, facilitating the future integration of external MPC resources and interactions with external agents.

- **Feedback**—User-agent interactions are anonymized, stored in a database, aggregated, and analyzed to evaluate possible corrective actions.

Use Cases

In this section, we describe common scenarios in which DAVE can be effectively employed. A live demonstration of the DAVE agent is openly available in the ColLab Virtual Research Environment https://services.d4science.org/web/collab/. This demo environment allows users to explore the main functionalities described in this paper, including interaction with selected D4Science services through a natural language interface.

Document Retrieval and Analysis

A common use case involves retrieving documents from the D4Science Workspace and conducting an analysis on them. For instance, a user may request a list of files stored in a specific folder and then select one of them for further inspection. Figures 4 and 5 illustrate this interaction: the system first returns the available documents and subsequently produces an analysis of the chosen one. In this case, it is a scientific paper.

In this workflow, the *VRE Assistant* interprets the user's request to access an underlying service and delegates it to the *Service Dispatcher Agent*. The *Service Dispatcher* determines that the request requires interaction with the Workspace and forwards it to the *Workspace Agent*, which performs the appropriate tool call. In the first request, the list of files is returned to the *VRE Assistant*, which presents it to the user. In the second case, once the user selects a document, the *VRE Assistant* updates the context and delegates

Fig. 5 Example user—DAVE interaction: document retrieval and analysis (cont.). The complete transcript is in Appendix A

USER: tell me more about the platform

DAVE: Of course. Here is some more information about the D4Science platform
D4Science (Data Infrastructure for Science) is a comprehensive, service-oriented, and hybrid data e-infrastructure [**GK**]. It is a digital infrastructure designed to offer diverse communities of practice a comprehensive suite of services through tailored and co-created virtual research environments (VREs), promoting collaboration and innovation [**KB**]. . . .
Key Concepts and Features

- Virtual Research Environments (VREs): The central concept of D4Science is the VRE, a web-based workspace tailored to the needs of a specific research community or project [**GK**]. D4Science delivers VREs as a Service, which are web-based, community-oriented, collaborative, and user-friendly working environments [**KB**].
- Open Science and FAIR Principles: The platform is a key enabler of Open Science, providing researchers with an environment where open practices are transparently promoted [**KB**]. It is designed to support the FAIR data principles: making data Findable, Accessible, Interoperable, and Reusable [**GK**].
- Reproducibility and Reusability: The platform is designed to enable the reproducibility and reusability of research processes [**KB**].
- . . .

Core Services and Frameworks
D4Science provides a rich portfolio of services and research products that can be combined to create custom VREs [**GK, KB**]. These include:

- Data Management and Storage: This includes a personal and shared Workspace and a Data Catalogue for publishing and discovering datasets [**GK**]. The platform provides a secure, efficient, and scalable Data Storage Framework [**KB**].
- . . .

Underlying Technology
The entire D4Science infrastructure is powered by the gCube System [**GK**]. gCube is an open-source software system that acts as the middleware for building and operating VREs, integrating distributed resources into a coherent set of services [**GK**].
Users and Applications
D4Science is used by a wide range of scientific communities, particularly in the environmental and life sciences, including fisheries, aquaculture, biodiversity, and marine and earth observation [**GK**]. . . .
Legend:
[**KB**]: Information from the Knowledge Base.
[**GK**]: Information from General Knowledge.

the task to the *Document Analyzer Agent*, which retrieves metadata and produces a summary of the document.

The last interaction (Fig. 5) involves the user asking for further information about D4Science, which is provided by the *VRE Assistant* with the assistance of the *Knowledge Retriever Agent*.

Social Posts Overview

Another common use case for DAVE is to provide an overview of recent activity in the VRE's social space. A user can ask the system to summarize posts published during a specific timeframe, such as the last seven days or a specific month, to quickly get an overview of recent discussions and shared resources. Figure 6 illustrates such an interaction, where the user's request is translated into a concise summary of the relevant posts. Then, the user keeps asking questions to get more precise contextual information.

In this workflow, the *VRE Assistant* receives the user's request and delegates it to the *Social Agent*. The Social Agent, in turn, requests relevant posts and comments from the *Knowledge Retriever Agent*. These posts are retrieved from the *RAG*, which is periodically populated and updated by a dedicated pre-fetcher job that fetches and processes

USER: Please recap what happened in the XXXXX VRE between january and march 2021

DAVE: Here's a recap of what happened in the XXXXX VRE between January and March 2021:

- **January 20, 2021**: The project brochure in English was uploaded to the VRE workspace. A discussion ensued about adapting its academic language for local stakeholders, with partners encouraged to modify the content for their local contexts.
- **February 18, 2021**: An invitation to the conference and exhibition (XXXXX) was shared, with the deadline for contributions extended to March 1, 2021.
- **February 23, 2021**: The official project website was launched, and partners were invited to visit and contribute news and events.
- **March 1, 2021**: An announcement was made for the XXXXXXX XXXXXXX XXXXXXX XXXXXXX virtual event, scheduled for March. Partners were advised to register by March 15.
- **March 4, 2021**: A news item about this meeting was published on the website, and partners were encouraged to share it.

USER: what about the XXXXXXXX XXXX XXXXX meeting

DAVE: The XXXXXXXX XXXX XXXXX (XXXXX XXXXX) meeting was announced by XXXXXXXX for XXXXXXXX XXXXXXXX and a Zoom link was provided. Some participants, XXXXXXXX XXXXXXXX and XXXXXXXX XXXXXXXX initially experienced issues accessing the workspace to download the agenda, which XXXXXXXX XXXXXXXX helped to resolve through a dedicated session. The access issue was fixed by March 4th.
Following the main SC meeting, a two-hour follow-up session was scheduled with a Doodle poll. A smaller preparatory meeting was also organized with XXXXXXXX XXXXXXXX, XXXXXXXX XXXXXXXX and XXXXXXXX.

After the SC meeting, XXXXXXXX XXXXXXXX highlighted two action points:

1. Each partner organization was requested to appoint a Communication Responsible for a new Communication Taskforce.
2. Partners were encouraged to share relevant news and events for the XXXXX website. Several partners confirmed their representatives for the task force.

A news item about the XXXXXXXX meeting was published on the XXXXXXXX website on March 4th, and partners were encouraged to share it.

Fig. 6 Example user—DAVE interaction: social posts summarization

social content. Once the posts are provided by the *Knowledge Retriever Agent*, the Social Agent directly summarizes this information to produce a coherent and concise overview of recent activity, allowing the user to stay informed with no need of reading each individual post. When asked about specific aspects of the posts, the flows retrieves this information through the *Knowledge Retriever Agent* and returns it to the user.

Catalogue Navigation

Another key use case for DAVE is navigating and summarizing the research items available in the VRE's Catalogue. Researchers often need to quickly assess the relevance of a dataset—as well as any other research objects published by it—without examining each metadata field manually. By leveraging DAVE, users can request summaries of specific catalogue items, ask for overviews of entire thematic

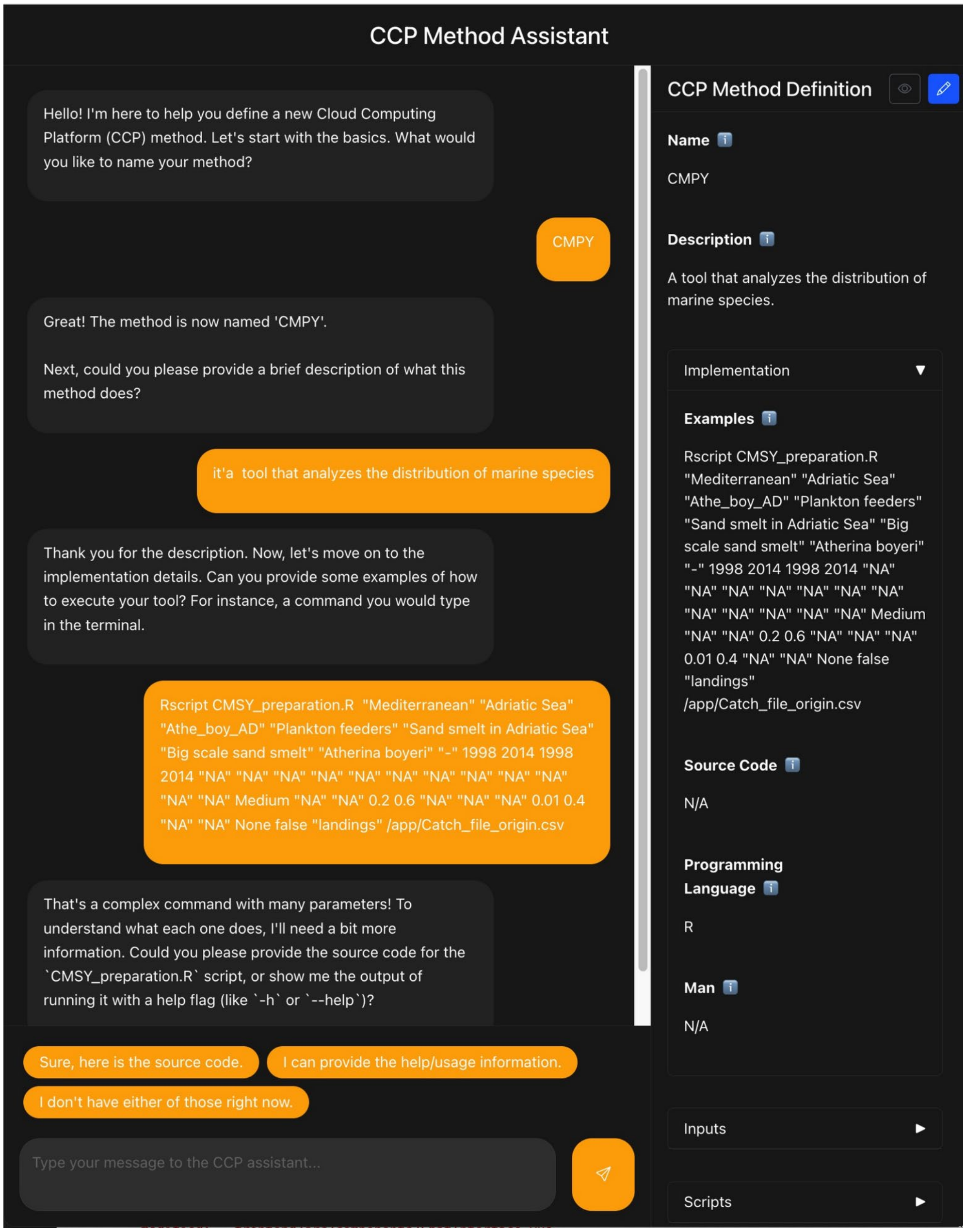

Fig. 7 Example interaction for CCP algorithm integration

clusters, or compare multiple datasets based on selected criteria such as size, license, or scope of use.

For instance, a user may ask the system to summarize a dataset of interest. In response, the system provides a coherent overview derived from the dataset's metadata, including its scope, contents, access conditions, and licensing terms. Alternatively, the user may request an overview of the entire Catalogue content.

In this workflow, the *VRE Assistant* delegates the user's request to the *Service Dispatcher Agent*, which forwards it to the *Catalogue Agent*, which acts as the interface to the D4Science Catalogue. The Catalogue Agent acts differently in the two cases proposed: in case of a single dataset, it retrieves it directly through the D4Science Python Library and extracts key information from it. In case of a general overview, it asks the *Knowledge Retriever Agent* to retrieve the relevant information from the *RAG*, which is populated by a pre-fetcher job similar to social data.

This functionality allows researchers to interact with the Catalogue conversationally, making resource discovery more intuitive and less time-consuming. Instead of navigating user interfaces or reading extensive metadata descriptions, users can obtain contextualized responses tailored to their needs, enabling faster and more informed decisions about dataset relevance and reuse.

Algorithm Integration with CCP

A significant use case for DAVE is to assist researchers in integrating their own algorithms as CCP methods within a VRE, particularly for users with limited programming experience or no prior knowledge of CCP method definition. This process involves guiding the user through the complex steps of preparing and deploying their code to execute on the D4Science infrastructure. A sample interaction is presented in Fig. 7.

The interaction begins on a dedicated web interface within the CCP section of the D4Science portal. This interface leverages DAVE to collect the necessary data to define the CCP method. DAVE guides the user in identifying critical input and output parameters, including details such as their mandatory status, data type, and default values. It further assists in selecting an appropriate Docker container for execution and in drafting the deployment and execution scripts for the method within that container.

To achieve this, DAVE actively interacts with the user, collecting provided data and analyzing the source code of the algorithm to be imported, or examples of its invocation. Concurrently, DAVE answers any user questions by accessing relevant information about CCP, Docker containers, and the system in general from its Knowledge Base, providing immediate, context-aware support throughout the integration process.

Conclusion and Future Work

This paper presents our experience in the development and deployment of conversational agents within the D4Science infrastructure, culminating in the design and implementation of DAVE, the current VRE-oriented agent that powers D4Science. We reported three successive implementation approaches, each building on the limitations and lessons learned from the preceding iteration, and guided by a set of generalizable requirements for conversational agents in VREs. Our analysis highlights the unique challenges of integrating conversational agents into Virtual Research Environments and Science Gateways, including: the need to support diverse workflows and tasks ranging from generic ones to context-specific ones; the ability to evolve alongside rapidly changing underlying infrastructure capabilities and user expectations; the capability to seamlessly evolve to leverage the development of AI-related technologies, including the exploitation of context-specific LLMs and generalist LLMs.

The resulting DAVE agent shows that conversational interfaces can provide an effective, user-friendly entry point to VRE functionalities, supporting both technical and non-technical users. Our experience suggests that such agents can lower the adoption barrier, streamline workflows, and foster greater engagement within research communities. They can act as adaptive collaborators, helping users navigate complex environments, automating routine tasks, and providing contextual guidance tailored to individual needs.

The future development of DAVE will follow a co-creation-oriented approach, actively involving representative users in the refinement process. An extensive user evaluation campaign is planned to systematically assess the agent's usability, effectiveness, and overall user satisfaction. This campaign will build on preliminary feedback collected in selected scenarios and combine quantitative metrics such as task completion time, interaction success rate, and error frequency, and qualitative insights into user experience and perceived usefulness.

Future work will also focus on extending the feedback mechanism implemented in DAVE with *qualitative* metrics. We plan to incorporate user-provided ratings and/or inferred satisfaction signals to guide further developments and refinements on the various functionalities.

In addition, we will work on integrating in DAVE a locally hosted LLM. Running the model within the same infrastructure will allow all processing to remain on-site, thereby strengthening data protection and confidentiality. This approach is expected not only to mitigate risks associated with transmitting sensitive information to external providers but also to reduce operational costs by limiting reliance on commercial LLM services.

Appendix A. Full Transcript of Example in Section Document Retrieval and Analysis

User: Can you list the content of my `papers` folder?

DAVE: Here are the contents stored in your folder:

- `Enacting_open_science.pdf`
- `Open_Science_Survey.pdf`

User: Please analyze `Enacting_open_science.pdf`.

DAVE: Here is a summary of the document `Enacting_open_science.pdf`:
Summary: This paper describes the D4Science platform, a technological infrastructure designed to support and enact the principles of Open Science. The authors argue that for the Open Science movement to succeed, it needs not only cultural shifts but also enabling technologies that make open practices easier for researchers. D4Science provides a suite of integrated services to create "Virtual Research Environments" (VREs) that facilitate collaboration, transparency, and reproducibility throughout the research lifecycle.
The core components of the D4Science offering include:

1. A shared workspace for storing, organizing, and versioning research artefacts.
2. A social networking platform for communication, discussion, and collaboration among researchers.
3. A data analytics platform for executing experiments and processing data, which captures provenance to ensure reproducibility.
4. A publishing platform (catalogue) that enables researchers to effectively find, access, cite, and reuse all types of research artefacts, not just traditional papers.

By integrating these tools, D4Science aims to create a seamless environment where research activities are transparently captured, making the entire process more fair, solid, and democratic.
Authors:

- M. Assante
- L. Candela
- D. Castelli
- R. Cirillo
- G. Coro
- L. Frosini
- L. Lelii
- F. Mangiacrapa
- P. Pagano
- G. Panichi
- F. Sinibaldi

Keywords:

- Open Science
- gCube
- D4Science
- Social Networking
- Analytics
- Publishing

USER: tell me more about the platform

DAVE: Of course. Here is some more information about the D4Science platform
D4Science (Data Infrastructure for Science) is a comprehensive, service-oriented, and hybrid data e-infrastructure [**GK**]. It is a digital infrastructure designed to offer diverse communities of practice a comprehensive suite of services through tailored and co-created virtual research environments (VREs), promoting collaboration and innovation [**KB**]. Its primary purpose is to provide these VREs to scientific communities, enabling them to conduct collaborative, data-intensive research [**GK**]. It is operated as a not-for-profit service and managed by the Italian National Research Council (CNR) [**KB**].
Key Concepts and Features

- Virtual Research Environments (VREs): The central concept of D4Science is the VRE, a web-based workspace tailored to the needs of a specific research

community or project [**GK**]. D4Science delivers VREs as a Service, which are web-based, community-oriented, collaborative, and user-friendly working environments [**KB**].
- Open Science and FAIR Principles: The platform is a key enabler of Open Science, providing researchers with an environment where open practices are transparently promoted [**KB**]. It is designed to support the FAIR data principles: making data Findable, Accessible, Interoperable, and Reusable [**GK**].
- Reproducibility and Reusability: The platform is designed to enable the reproducibility and reusability of research processes [**KB**].
- Integration and Interoperability: A major goal is to break down data silos by integrating a vast array of data sources, computational services, and software from different providers into a seamless environment [GK]. D4Science integrates multiple tools into cohesive, user-friendly services, accessible to researchers with varying technical expertise [**KB**].
- Co-creation Approach: D4Science utilizes a co-creation approach to ensure that the resulting VREs are well-matched to the expectations of their designated communities [**KB**].

Core Services and Frameworks
D4Science provides a rich portfolio of services and research products that can be combined to create custom VREs [**GK, KB**]. These include:

- Data Management and Storage: This includes a personal and shared Workspace and a Data Catalogue for publishing and discovering datasets [**GK**]. The platform provides a secure, efficient, and scalable Data Storage Framework [**KB**].
- Data Processing and Analytics: The platform provides on-demand computing, an Algorithm Importer for custom scripts, and a Data Miner for complex workflows [**GK**]. It also offers a Data Analytics Framework to support advanced data analysis across various scientific domains [**KB**].
- Collaboration and Communication: Features include shared workspaces, social networking tools, and collaborative document management [**GK**]. A Community Framework fosters collaboration and knowledge sharing within research communities [**KB**].
- Publishing and Dissemination: Tools are available to formally publish datasets, often with a Digital Object Identifier (DOI) [**GK**]. The Data Publishing Framework streamlines the publication, discovery, and management of data, methods, libraries, and tools [**KB**].
- Software and Service Management: D4Science includes a Software Framework to support the development and management of software projects and a Service Framework for deploying and managing applications [**KB**].

Underlying Technology

The entire D4Science infrastructure is powered by the gCube System [**GK**]. gCube is an open-source software system that acts as the middleware for building and operating VREs, integrating distributed resources into a coherent set of services [**GK**].
Users and Applications
D4Science is used by a wide range of scientific communities, particularly in the environmental and life sciences, including fisheries, aquaculture, biodiversity, and marine and earth observation [**GK**]. The platform has a consistently growing user base and has hosted over 440,000 working sessions in the last five years, with an average of about 6,800 working sessions per month [**KB**].
Legend:
[**KB**]: Information from the Knowledge Base.
[**GK**]: Information from General Knowledge.

Acknowledgements The authors would like to thank Ahmed Salah Tawfik Ibrahim for valuable contributions to the design and development of Janet.

Author Contributions Massimiliano Assante: Writing—review & editing; Leonardo Candela: Supervision, Writing—original draft, Writing—review & editing; Andrea Dell'Amico: Resources, Writing—review & editing; Luca Frosini: Project administration, Writing—review & editing; Francesco Mangiacrapa: Writing—review & editing; Alfredo Oliviero: Conceptualization, Investigation, Methodology, Software, Writing—original draft; Pasquale Pagano: Project administration, Supervision, Writing—review & editing; Giancarlo Panichi: Writing—review & editing; Biagio Peccerillo: Conceptualization, Investigation, Methodology, Software, Writing—original draft; Tommaso Piccioli: Resources, Writing—review & editing; Marco Procaccini: Resources, Writing—review & editing.

Funding Open access funding provided by Consiglio Nazionale Delle Ricerche (CNR) within the CRUI-CARE Agreement. This work has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement N. 101094227 (Blue-Cloud 2026). This work is partially supported by the projects: "FOSSR—Fostering Open Science in Social Science Research" (IR0000008)—CUP B83C22003950001, funded by the European Union, NextGenerationEU; "ITINERIS—Italian Integrated Environmental Research Infrastructures System" (IR0000032)—CUP B53C22002150006, funded by the European Union, NextGenerationEU.

Data Availability Not applicable.

Materials Availability Not applicable.

Code Availability The source code developed and used in this study is openly available under EUPL−1.1-or-later license in the Gitea repository at the following links. *Janet*: https://code-repo.d4science.org/D4Science/Janet, https://code-repo.d4science.org/D4Science/JanetFrontEnd, https://code-repo.d4science.org/D4Science/JanetBackEnd, https://code-repo.d4science.org/D4Science/JanetKnowledgeService, https://code-repo.d4science.org/D4Science/models. *D4Science AI Agent*: https://code-repo.d4science.org/D4Science/d4science_cheshirecat.git, https://code-repo.d4science.org/D4Science/d4science_cheshirecat_plugin.git. *DAVE*: https://code-repo.d4science.org/D4Science/dave_agent.

Declarations

Conflict of interest The authors have no Conflict of interest to declare that are relevant to the content of this article.

Ethical Approval and consent to participate Not applicable.

Consent for Publication Not applicable.

References

1. Xi Z, Chen W, Guo X, He W, Ding Y, Hong B, et al. The rise and potential of large language model based agents: a survey. Sci China Inf Sci. 2025;68(2):121101. https://doi.org/10.1007/s11432-024-4222-0.
2. Wang L, Ma C, Feng X, Zhang Z, Yang H, Zhang J, et al. A survey on large language model based autonomous agents. Front Comp Sci. 2024;18(6):186345. https://doi.org/10.1007/s11704-024-40231-1.
3. Kusal S, Patil S, Choudrie J, Kotecha K, Mishra S, Abraham A. AI-based conversational agents: a scoping review from technologies to future directions. IEEE Access. 2022;10:92337–56. https://doi.org/10.1109/ACCESS.2022.3201144.
4. Candela L, Castelli D, Pagano P. Virtual research environments: an overview and a research agenda. Data Sci J. 2013;12:75–81.
5. Lawrence KA, Zentner M, Wilkins-Diehr N, Wernert JA, Pierce M, Marru S, et al. Science gateways today and tomorrow: positive perspectives of nearly 5000 members of the research community. Concurr Comput Pract Exp. 2015;27(16):4252–68. https://doi.org/10.1002/cpe.3526 (https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.3526).
6. Gesing S, Pierce M, Marru S, Zentner M, Huff K, Bradley S, et al. Chap. 4. Science gateways and AI/ML: how can gateway concepts and solutions meet the needs in data science? In: Pietro AD, Martì J, editors., et al., Critical infrastructure. Rijeka: IntechOpen; 2023. https://doi.org/10.5772/intechopen.110144.
7. Sperhac J, Gesing S, Zentner M, Stirm C, Quick R, Stubbs J. AI and science gateways: a promising combination for accelerating science and research computing. In: Practice and experience in advanced research computing 2024: Human Powered Computing. PEARC '24. Association for Computing Machinery, New York, NY, USA; 2024. https://doi.org/10.1145/3626203.3670562.
8. Assante M, Candela L, Castelli D, Cirillo R, Coro G, Frosini L, et al. Enacting open science by D4Science. Futur Gener Comput Syst. 2019;101:555–63. https://doi.org/10.1016/j.future.2019.05.063.
9. Assante M, Candela L, Castelli D, Cirillo R, Coro G, Dell'Amico A, et al. Virtual research environments co-creation: the D4Science experience. Concurr Comput Pract Exp. 2023;35(18):6925. https://doi.org/10.1002/cpe.6925 (https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.6925).
10. Candela L, Castelli D, Pagano P. The D4Science experience on virtual research environment development. Comput Sci Eng. 2023;25(2):12–9. https://doi.org/10.1109/MCSE.2023.3290433.
11. Ibrahim AST, Candela L. Introducing janet: early findings on a conversational agent for virtual research environments. In: Pagé C, Gesing S, editors. Proceedings of the 16th International Workshop on Science Gateways. 2024. https://doi.org/10.5281/zenodo.13862920
12. Oliviero A, Peccerillo B, Procaccini M. An LLM-powered Agent for the D4Science Digital Infrastructure. Technical Report ISTI-TR-2025/010, Istituto di Scienza e Tecnologie dell'Informazione "Alessandro Faedo" – ISTI, Consiglio Nazionale delle Ricerche. 2025. https://doi.org/10.32079/ISTI-TR-2025/010.
13. Cheshire Cat AI. Cheshire Cat AI. GitHub. 2025. https://github.com/cheshire-cat-ai. Accessed 11 Sep 2025.
14. Cheshire Cat AI. Cheshire Cat AI—production ready AI agent framework. 2025. https://cheshirecat.ai/. Accessed 11 Sep 2025.

15. Google. Agent Development Kit. 2025. https://google.github.io/adk-docs/. Accessed 11 Sep 2025.
16. Ibrahim AST, Candela L. Conversational agents for virtual research environments: a survey of the literature. ISTI Technical Report ISTI-2023-TR/007, ISTI-CNR. 2023. https://doi.org/10.32079/ISTI-TR-2023/007. Accessed 12 Apr 2024.
17. D4Science Team. D4Science—accelerating Collaboration and Innovation. 2025. https://www.d4science.org/. Accessed 18 Sep 2025.
18. Assante M, Candela L, Castelli D, Cirillo R, Coro G, Frosini L, et al. The gCube system: delivering virtual research environments as-a-service. Futur Gener Comput Syst. 2019;95:445–53. https://doi.org/10.1016/j.future.2018.10.035.
19. Assante M, Lettere M, Oliviero A, Pagano P. Empowering collaborative and reproducible large-scale data analytics with d4science. ERCIM News. 2025;2025(140).
20. D4Science Team. D4Science Thematic Gateways. 2025. https://services.d4science.org/thematic-gateways. Accessed 18 Sep 2025.
21. Minaee S, Mikolov T, Nikzad N, Chenaghlu M, Socher R, Amatriain X, Gao J. Large language models: a survey. 2024. arXiv:2402.06196.
22. Naveed H, Khan AU, Qiu S, Saqib M, Anwar S, Usman M, et al. A comprehensive overview of large language models. ACM Trans Intell Syst Technol. 2025;16(5):1–72. https://doi.org/10.1145/3744746.
23. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I. Attention is all you need. 2023. arXiv:1706.03762.
24. OpenAI Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, Almeida D, Altenschmidt J, Altman S, et al. GPT-4 Technical Report. 2024. arXiv:2303.08774.
25. OpenAI. GPT-5 System Card. 2025. https://cdn.openai.com/gpt-5-system-card.pdf. Accessed 11 Sep 2025.
26. OpenAI. Introducing ChatGPT. 2022. https://openai.com/index/chatgpt/. Accessed 11 Sep 2025.
27. Chowdhery A, Narang S, Devlin J, Bosma M, Mishra G, Roberts A, Barham P, Chung HW, Sutton C, Gehrmann S, et al. PaLM: scaling language modeling with pathways. 2022. arXiv:2204.02311.
28. Google. Gemini. 2025. https://gemini.google.com/app. Accessed 11 Sep 2025.
29. Dubey A, Jauhri A, Pandey A, Kadian A, Al-Dahle A, Letman A, Mathur A, Schelten A, Yang A, Fan A, et al. The llama 3 herd of models. 2024. arXiv:2407.21783.
30. Meta. Introducing Code Llama, a state-of-the-art large language model for coding. 2023. https://ai.meta.com/blog/code-llama-large-language-model-coding/. Accessed 11 Sep 2025.
31. Anthropic. Introducing the next generation of Claude. 2025. https://www.anthropic.com/news/claude-3-family. Accessed 11 Sep 2025.
32. Bai Y, Kadavath S, Kundu S, Askell A, Kernion J, Jones A, Chen A, Goldie A, Mirhoseini A, McKinnon C, Chen C, Olsson C, Olah C, Hernandez D, Drain D, Ganguli D, Li D, Tran-Johnson E, Perez E, Kerr J, Mueller J, Ladish J, Landau J, Ndousse K, Lukosuite K, Lovitt L, Sellitto M, Elhage N, Schiefer N, Mercado N, DasSarma N, Lasenby R, Larson R, Ringer S, Johnston S, Kravec S, Showk SE, Fort S, Lanham T, Telleen-Lawton T, Conerly T, Henighan T, Hume T, Bowman SR, Hatfield-Dodds Z, Mann B, Amodei D, Joseph N, McCandlish S, Brown T, Kaplan J. Constitutional AI: harmlessness from AI feedback. 2022. arXiv:2212.08073.
33. Zhang Q, Ding K, Lv T, Wang X, Yin Q, Zhang Y, et al. Scientific large language models: a survey on biological & chemical domains. ACM Comput Surv. 2025;57(6):1–38. https://doi.org/10.1145/3715318.
34. Adamopoulou E, Moussiades L. Chatbots: history, technology, and applications. Mach Learn Appl. 2020;2:100006. https://doi.org/10.1016/j.mlwa.2020.100006.
35. Chase H. LangChain. 2022. https://github.com/langchain-ai/langchain. Accessed 11 Sep 2025.
36. Qdrant. High-Performance Vector Search at Scale—Powering the next generation of AI applications with advanced, open-source vector similarity search technology. 2025. https://qdrant.tech/. Accessed 11 Sep 2025.
37. Qdrant. Qdrant Documentation. 2025. https://qdrant.tech/documentation/overview/. Accessed 11 Sep 2025
38. Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. Adv Neural Inf Process Syst. 2020;33:9459–74.
39. Sinha K, Singh V, Vishnoi A, Madan P, Shukla Y. Healthcare diagnostic RAG-based chatbot triage enabled by BioMistral-7B. In: 2024 International Conference on Emerging Technologies and Innovation for Sustainability (EmergIN); 2024. pp. 333–338. https://doi.org/10.1109/EmergIN63207.2024.10961136.
40. Vinay K, Srenik G, Krishnaveni N . Dynamic music composition system using LSTM and transformer networks. In: 2025 Fourth International Conference on Smart Technologies, Communication and Robotics (STCR); 2025. pp. 1–6. https://doi.org/10.1109/STCR62650.2025.11020414
41. Mehul K, Kanagavalli VR, Saradha KR, Gowtham PN, Sachin MP, Surya U, Godhandaraman R, Girish S, Naveen R. Gen AI driven FAQ chatbot using advanced RAG architecture for querying annual reports. In: 2025 International Conference on Computing and Communication Technologies (ICCCT); 2025. pp. 1–6. https://doi.org/10.1109/ICCCT63501.2025.11020025
42. Google Developers. Getting Started with MCP, ADK and A2A. 2024. https://codelabs.developers.google.com/codelabs/currency-agent. Accessed 16 Sep 2025.
43. Sebastián Ramírez. Typer. 2025. https://typer.tiangolo.com. Accessed 11 Sep 2025.
44. Hosseini S, Seilani H. The role of agentic AI in shaping a smart future: a systematic review. Array. 2025;26:100399. https://doi.org/10.1016/j.array.2025.100399.
45. Krishnan N. AI agents: evolution, architecture, and real-world applications. 2025. arXiv:2503.12687.
46. Yu X, Xia J, Shen Z. Artificial intelligence for research (AI4R): knowledge base and impact. In: 2024 International Conference on Intelligent Education and Intelligent Research (IEIR); 2024. pp. 1–8. https://doi.org/10.1109/IEIR62538.2024.10959986.
47. Gueroudji A, Mallick T, Souza R, Silva RF, Ross R, Dorier M, Carns P, Chard K, Foster I. ControlA: agentic workflow control mechanisms for reliable science. In: IEEE International Conference on e-Science; 2025. pp. 1–11. https://doi.org/10.13140/RG.2.2.11982.52800.
48. Gottweis J, Weng W-H, Daryin A, Tu T, Palepu A, Sirkovic P, Myaskovsky A, Weissenberger F, Rong K, Tanno R, Saab K, Popovici D, Blum J, Zhang F, Chou K, Hassidim A, Gokturk B, Vahdat A, Kohli P, Matias Y, Carroll A, Kulkarni K, Tomasev N, Guan Y, Dhillon V, Vaishnav ED, Lee B, Costa TRD, Penadés JR, Peltz G, Xu Y, Pawlosky A, Karthikesalingam A, Natarajan V. Towards an AI co-scientist. 2025. arXiv:2502.18864.
49. Poon SS, Thomas RC, Aragon CR, Lee B. Context-linked virtual assistants for distributed teams: an astrophysics case study. In: Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work. CSCW '08, pp. 361–370. Association for Computing Machinery, New York, NY, USA. 2008. https://doi.org/10.1145/1460563.1460623.
50. Lee L-K, Fung Y-C, Pun Y-W, Wong K-K, Yu MT-Y, Wu N-I. Using a multiplatform chatbot as an online tutor in a university course. In: 2020 International Symposium on Educational

Technology (ISET); 2020. pp. 53–56. https://doi.org/10.1109/ISET49818.2020.00021.
51. Yang K, Niu L. Framework design of a multi-educational-agent system for university lecturers. In: 2024 International Conference on Intelligent Education and Intelligent Research (IEIR); 2024. pp. 1–6. https://doi.org/10.1109/IEIR62538.2024.10960068.
52. Gao S, Fang A, Huang Y, Giunchiglia V, Noori A, Schwarz JR, et al. Empowering biomedical discovery with AI agents. Cell. 2024;187(22):6125–51. https://doi.org/10.1016/j.cell.2024.09.022.
53. Harrer S, Rane RV, Speight RE. Generative AI agents are transforming biology research: high resolution functional genome annotation for multiscale understanding of life. EBioMedicine. 2024;109:105446. https://doi.org/10.1016/j.ebiom.2024.105446.
54. He J, Treude C, Lo D. LLM-based multi-agent systems for software engineering: literature review, vision, and the road ahead. ACM Trans Softw Eng Methodol. 2025;34(5):1–30. https://doi.org/10.1145/3712003.
55. Zhang Y, Wei C, He Z, Yu W. Geogpt: an assistant for understanding and processing geospatial tasks. Int J Appl Earth Obs Geoinf. 2024;131:103976. https://doi.org/10.1016/j.jag.2024.103976.
56. Sanh V, Debut L, Chaumond J, Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. 2020. arXiv:1910.01108.
57. Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. RoBERTa: a robustly optimized BERT pretraining approach. 2019. arXiv:1907.11692.
58. Song K, Tan X, Qin T, Lu J, Liu T-Y. MPNet: masked and permuted pre-training for language understanding. 2020. arXiv:2004.09297.
59. Bajaj P, Campos D, Craswell N, Deng L, Gao J, Liu X, Majumder R, McNamara A, Mitra B, Nguyen T, Rosenberg M, Song X, Stoica A, Tiwary S, Wang T. MS MARCO: a human generated MAchine Reading COmprehension Dataset. 2018. arXiv:1611.09268.
60. Jin Q, Dhingra B, Liu Z, Cohen WW, Lu X. PubMedQA: a dataset for biomedical research question answering. 2019. arXiv:1909.06146.
61. Dasigi P, Lo K, Beltagy I, Cohan A, Smith NA, Gardner M. A dataset of information-seeking questions and answers anchored in research papers. 2021. arXiv:2105.03011.
62. Lu P, Mishra S, Xia T, Qiu L, Chang K-W, Zhu S-C, et al. Learn to explain: multimodal reasoning via thought chains for science question answering. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A, editors. Advances in neural information processing systems, vol. 35. Red Hook: Curran Associates Inc.; 2022. p. 2507–21.
63. Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res. 2020;21(1):5485–551.
64. Ye J, Chen X, Xu N, Zu C, Shao Z, Liu S, Cui Y, Zhou Z, Gong C, Shen Y, Zhou J, Chen S, Gui T, Zhang Q, Huang X. A comprehensive capability analysis of GPT-3 and GPT-3.5 series models. 2023. arXiv:2303.10420.
65. Johnson J, Douze M, Jégou H. Billion-scale similarity search with GPUs. IEEE Trans Big Data. 2021;7(3):535–47. https://doi.org/10.1109/TBDATA.2019.2921572.
66. Boonstra L. Prompt Engineering. Google, Mountain View, CA; 2024. https://www.kaggle.com/whitepaper-prompt-engineering.