E3 Atlas: An Autonomous LLM Agent for Curating and Serving a Departmental Knowledge Base

Drejc Pesjak*
drejc.pesjak@ijs.si
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Georgi Trajkov*
geotrajkov0@gmail.com
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Luis Rei
luis.rei@ijs.si
Jožef Stefan Institute
Ljubljana, Slovenia

Dunja Mladenić
dunja.mladenic@ijs.si
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Abstract

Institutional knowledge in research organizations is fragmented across shared folders, project archives, publication lists, and the memories of individual researchers. We present E3 Atlas, a deployed system in which a large language model (LLM) agent builds, verifies, and serves a knowledge base for the Department of Artificial Intelligence (E3) at the Jožef Stefan Institute. The agent runs on the open-source Hermes harness and combines several memory layers, the main one being a typed Markdown knowledge graph of people, projects, organizations, publications, and resources. Graph writes follow a verify-before-write methodology with per-node confidence and provenance, and every node is approved by a human. On a 485-question benchmark, Atlas far outperforms an ungrounded LLM (78.4% vs. 20.0% correct) but does not beat simpler grounded baselines: classical RAG reaches 80.6% and agentic file search 81.6%. On 50 questions whose answers the corpus does not support, Atlas gave a confident answer 26 times, while RAG declined all 50. We describe the architecture, the curation methodology, and the evaluation, and conclude that verify-before-write must be matched by verify-before-answer.

Keywords

LLM agents, knowledge graphs, knowledge management, retrieval-augmented generation, human-supervised curation

1 Introduction

Research departments accumulate knowledge faster than anyone documents it: staff and roles change, projects start and finish, and deliverables, papers, tools, models, and recorded talks are produced continuously. This knowledge is fragmented across shared folders, grant archives, mailing lists, and personal recollection. Newcomers onboard slowly, and simple questions ("who worked on emotion detection?", "which Horizon Europe projects are active?") require asking the right person rather than querying a system.

We present **E3 Atlas**, a deployed system that realizes this pattern for the Department of Artificial Intelligence (E3) of the Jožef Stefan Institute. A single always-on LLM agent maintains the department's knowledge base: a typed Markdown knowledge graph, prose summaries, and recurring intelligence briefings, all served to department members through an authenticated web portal and a Slack front end. Our contributions are:

(1) the architecture of a practical, multi-layer-memory agentic knowledge management system with built-in automation;
(2) a verify-before-write methodology with provenance tracking that keeps agent-written knowledge auditable.

2 Related Work

Retrieval-augmented generation (RAG) [5] addresses the *query-time* half of the information retrieval problem. GraphRAG [2] extends classical RAG by first building an entity graph over a corpus to better support global, query-focused summarization. RAG answers from whatever corpus it is given, while keeping that corpus complete, current, and consistent is usually left to humans. Recent work on LLM agents [10, 9] suggests a different division of labor: let an autonomous agent do the continuous curation, including collecting evidence, verifying facts, and writing structured knowledge, while humans supervise and correct. This is the approach E3 Atlas takes: corpus *construction and maintenance* is the agent's primary task, with retrieval as a downstream benefit.

We use a Knowledge Graph (KG) as an organizational substrate [3]. Our graph is deliberately lightweight, inspired by the LLM Wiki concept [4]: typed Markdown files with YAML frontmatter under version-controlled review, favoring human-readable, directly editable representations over formal semantics and specialized tooling.

The layered-memory design of E3 Atlas is influenced by agent memory architectures such as MemGPT [7] and mechanisms surveyed in [11]. The agent skills mechanism follows the emerging practice of agents maintaining procedural knowledge [1].

3 System Architecture

3.1 Agent Harness and Models

Figure 1 gives an overview of the system. E3 Atlas runs on *Hermes* [6], an open-source autonomous agent harness, deployed on a dedicated department virtual machine. The harness provides the tool loop (terminal, file manipulation, web search and extraction, sub-agent delegation), a multi-platform gateway (Slack, CLI, web

*Both authors contributed equally to this research.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
Information Society 2026, Ljubljana, Slovenia
© 2026 Copyright held by the owner/author(s).
https://doi.org/10.70314/is.2026.sikdd.90

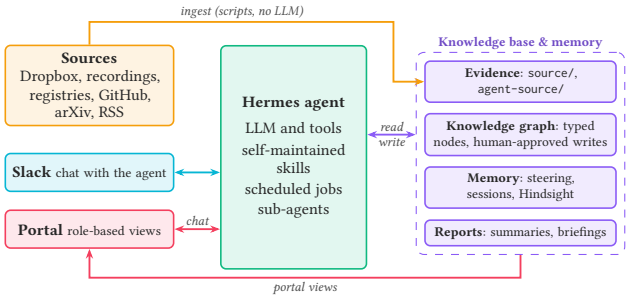

Figure 1: E3 Atlas architecture.

dashboard), scheduled jobs, and persistent state. The model used is Anthropic Claude, with an automatic fallback chain through OpenAI GPT and xAI Grok so that scheduled jobs do not stall on outages. On top of the stock harness we added the knowledge-base layout and its curation rules, the knowledge-graph schema, the memory layers, the briefing pipelines, the department portal, and all the related agent skills.

3.2 Knowledge Base Layout

The knowledge base separates evidence from derived knowledge:

- `source/` holds human-provided authoritative material (read-only to the agent);
- `source-text/` holds the text extracted from the provided documents (read-only to the agent);
- `agent-source/` holds raw evidence the agent collects (web pages, papers, external repositories);
- `knowledge-graph/` holds the durable entity nodes;
- `summaries/` and `briefings/` hold one-off reports and recurring intelligence products generated by the agent.

This separation helps make every derived claim traceable to evidence the agent did not author - a precondition for auditing machine-written knowledge.

3.3 The Knowledge Graph

The knowledge graph (KG) is a typed, interlinked corpus of plain Markdown files with the filesystem as the store, kept under Git version control: one file per node, folder-partitioned by node type. The following node types exist: person, project, publication, organization, meeting, infrastructure, programme, and resource with subtypes that include code repositories, datasets, models, and tools. Nodes are authored by an LLM agent that reads from the knowledge base source data folders (e.g. `source/`), or from sanctioned external registries, and writes claims with their associated provenance. Each file starts with YAML frontmatter: a machine-readable key-value block that includes:

- a globally unique slug, serving as the identifier for the node;
- relationship fields, with keys named for the relation itself (e.g. authors, funder, partners) and slugs of other nodes as the values;
- external identifiers (e.g. DOI, ORCID, CORDIS, URLs);
- sources (where claims came from);
- subject-descriptor field containing lists of topics or keywords.

The schema restricts which relationships each node type may carry. Most must resolve to an existing node, but a few may name people or organizations that deliberately get no node, such as external collaborators. Sources are paths to read-only evidence, registry URLs and identifiers, or plain-text references to confirmation by the administrator.

Subject-descriptor fields (topics, expertise, or keywords, depending on type) draw on a controlled vocabulary of research domains and methods declared in the schema, and the validator flags unregistered terms. Descriptors that the transcript pipeline emits for meeting nodes are free text.

After the frontmatter comes a short, type-specific prose body (typically 100–400 words) for human readers: the author abstract for publications, the funder's objectives for projects, a description plus hard facts (repository, licence, activity) for code, datasets, and models, and for people a summary of role and research focus with their national research-registry record, timestamped at retrieval.

Edges take two forms. *Hard links* are typed, directed slugs in the frontmatter, each written once on the side whose list is bounded and authoritative (a publication lists its authors and parent project, and a resource lists its maintainers). Inverse directions come from computed backlinks. *Soft links* are untyped `[[wikilinks]]` in the prose body. A static-site generator publishes the KG as a web viewer with backlinks and a graph view, and a hybrid lexical/vector index makes it searchable together with the underlying knowledge base.

3.4 Memory Layers

Four complementary memory layers keep the agent effective across sessions:

(1) *steering memory* provides a compact size-limited dynamic set of instructions, conventions, preferences, and behavioral rules that is injected into every turn;
(2) *session store*, a searchable archive of past conversations;
(3) *Hindsight* [8], a cross-session entity-centric memory backed by a relational store;
(4) *agent skills*, a set of self-authored and self-maintained procedural documents (e.g., how to compile papers in the IS template, how to query the national research registry, how to curate the knowledge graph).

Durable, curated departmental knowledge belongs in the knowledge graph, accumulated entity-centric knowledge in Hindsight, procedures in skills, and only compact behavioral rules and preferences in the capped steering memory.

3.5 Automation and Human Oversight

Automation differs by task. *Ingestion* of new data and maintenance run automatically as scheduled scripts that invoke no model. *Graph writes* are generated by the model but approved by a human, and *briefings* are fully autonomous.

Ingestion mirrors external sources such as Dropbox, converts PDF, slide, and Word documents to Markdown, and refreshes the retrieval index. It also searches public repositories (e.g., Zenodo, Hugging Face) and bibliographic indexes, and checks that each resource was authored by a department member under department funding. This makes new material *searchable*. The model-driven step that reads imported material and writes nodes is a separate job that a human starts, and every node it writes is approved by a human.

Maintenance jobs are script-only and follow a watchdog convention: silent on success, reporting only on change or failure. The simple example is the nightly automated backup of the knowledge graph as a compressed archive.

Briefings are the fully autonomous category. Six topic-scoped jobs run overnight to produce daily AI research and ecosystem briefings from sources including arXiv, X, RSS feeds, and blog posts. Their top-ranked items are aggregated into a single daily briefing delivered to the department Slack channel.

3.6 Web Interface

An authenticated web portal with role-based access gives department members an entry point to the knowledge graph, meeting transcripts and recordings, briefings, and generated summaries, plus a chat interface to the agent and administrative views of the agent and its memory. Views read the agent-maintained knowledge base directly, so new content appears without a separate publication step.

4 Curation Methodology

Letting an LLM write an institutional record raises an obvious risk: confident fabrication. The E3 Atlas methodology treats this as the central design constraint.

4.1 Verify before write and Provenance

A node may be minted only after its key facts are confirmed by authoritative evidence (publication, institutional page, project database, national research registry). A single authoritative source is the minimum requirement, but claims are cross-checked against multiple independent sources when possible. When evidence is partial, the agent must mint a stub with *'confidence: low'* and a *'needs-review'* tag rather than guess.

Every node lists its sources, derived claims in prose link back to evidence files or URLs. When sources disagree, a precedence rule decides. For example, a signed grant agreement overrides the proposal and its evaluation report on titles, consortium composition, and deliverables.

4.2 Schema governance

The schema itself evolves under explicit, numbered rules negotiated with human curators — e.g., a rule fixing how partner organizations are modeled (the participating unit becomes the node, and external contacts are kept inline rather than as person nodes, partly for data minimization). Rules are recorded in the schema document and enforced by audit scripts that check frontmatter validity, slug uniqueness, and dangling edges.

4.3 Corrections

Department members who identify gaps or mistakes can correct the KG through the chat interface, a process of **Human-in-the-loop correction** which requires the agent to find/obtain and verify supporting evidence. When these corrections reveal a procedural gap, the relevant agent skill is potentially semi-automatically improved so the same mistake is not repeated.

Automatic correction covers errors that scripts can detect and often fix deterministically, such as Markdown syntax errors, missing fields like ORCID, and missing or invalid provenance. These scripts are invoked by agent skills or listed in `AGENTS.md`, and keep the graph valid and free of dangling links.

5 Evaluation

5.1 Evidence Corpus and Knowledge Graph

Table 1 lists the largest document categories in the evidence corpus and the largest node types in the knowledge graph at the evaluation snapshot. The agent can additionally access 284 past conversations and 20,768 memory units in Hindsight.

Table 1: Top evidence-corpus categories by file count (left) and top knowledge-graph node types (right).

Corpus category	Files
Meeting/seminar transcript	292
Presentation	90
Proposal	89
Grant/consortium agreement	78
Evaluation	73
Deliverable	58

Node type	Nodes
Meeting	297
Publication	169
Project	153
Repository	114
Organization	77
Person	58

5.2 QA Benchmark

We sampled evidence passages across source types and document sizes, and an LLM (GPT-5.6-terra) wrote a question, gold answer, and evidence reference for each. Each type targets one capability. **Single-fact** (F) questions test retrieving one fact from one passage. **List/aggregation** (L) questions test collecting several items, usually within one document. **Relational** (R) questions test joining facts from two documents that share an entity. **Negative** (N) questions ask about plausible relations the corpus does not support, testing abstention (absence was checked against the retrieval index). A separate LLM verifier and a programmatic answer-leak check removed questions answerable without the corpus, ambiguous, or revealing their answer. Of 587 candidates, 487 were kept and 485 were judged for every system.

An author audit of 20 question–answer pairs found one incorrect gold answer (5%, 95% Wilson interval 0.9–23.6%), which was not corrected before scoring. Many questions reuse the vocabulary and identifiers of their source passage, so they do not reflect how people phrase questions.

Scoring. An LLM judge (GPT-5.6-sol) saw the question, gold answer, and system answer, but not the evidence, and scored 2 if all key facts were present and none contradicted, 1 if partially correct (e.g., an incomplete list), and 0 otherwise, ignoring phrasing and extra detail. Only 2 counts as correct. On negative questions a refusal scores 2 and any confident assertion 0. Such an assertion is unsupported by the corpus but not necessarily false. An independent model agreed with the judge on 20 of 20 sampled answers, but this check was not blind, and the judge shares a model family with the answering model.

5.3 Setup and Results

All systems answered with GPT-5.6-terra (temperature 1, fixed seed), without web access, once per question. The ungrounded LLM answered closed-book. Classical RAG used structure-aware, deduplicated 500-token chunks, hybrid BM25 and BGE-M3 retrieval with reciprocal rank fusion, and the top eight chunks. Agentic file search ran a fresh read-only Codex sandbox per question with ripgrep and sed. E3 Atlas ran in the Hermes harness (v0.20.5) with terminal, file, and memory tools and a 420 s limit. For Atlas, the graph was pinned to a fixed commit and the corpus to a fixed hash, and their fingerprint and that of the memory store were identical before and after the run. Memory writes and session search were disabled so nothing carried over between questions, and a tool gate blocked all 12 attempts to read benchmark or gold files.

Table 2 reports accuracy by question type. We compared systems with the two-sided exact McNemar test, which suits paired binary outcomes and uses only the questions on which two systems disagree, and applied a Holm correction to the three comparisons involving Atlas. On the 435 answerable questions, Atlas clearly outperformed the ungrounded baseline (adjusted $p < .001$). It did not differ significantly from RAG ($p = .121$) but was below agentic file search ($p = .019$, adjusted $p = .038$). Ten false refusals by Atlas account for most of the questions that only file search answered. Over all 485 questions Atlas ranks last of the three grounded systems, though neither difference is significant (RAG $p = .36$, file search $p = .052$). Whole-file access helped most on relational questions, where both agentic systems lead RAG by a wide margin.

Table 2: QA accuracy by question type (correct = judge score 2).

System (n)	F 237	L 98	R 100	N 50	Answ. 435	All 485
Ungrounded LLM	.169	.031	.160	.760	.136	.200
Classical RAG	**.924**	.643	.590	**1.000**	.784	.806
Agentic file search	.903	**.735**	**.880**	.440	**.860**	**.816**
E3 Atlas	.861	.704	.840	.460	.821	.784

RAG declined all 50 negative questions. Atlas declined 23, and 26 of its 27 other answers were confident, unhedged assertions, a pattern file search shares (26 of 28). The additional knowledge layers were rarely used: of the 426 answerable questions Atlas did not refuse, only 33 cited a knowledge-graph file, with similar accuracy with and without it (0.848 vs. 0.840), and the memory tool was never called although it was available. Reliable deployment therefore requires an answering procedure that specifies how to use the graph and memory, verifies provenance, and checks for absent evidence.

6 Discussion, Limitations, and Future Work

Why the approach is practical. E3 Atlas builds on information that the department already produces: project records in CORDIS and SICRIS, researcher profiles, publications, models, recorded meetings, and internally shared proposals. Much of this information is structured and accessible through databases or APIs, while the remaining material is generated by existing workflows. Consequently, deployment requires little change in staff behaviour: the agent performs the otherwise manual work of collecting, linking, and updating knowledge.

Trustworthiness and answering. Keeping evidence apart from derived knowledge, recording provenance and confidence, flagging uncertain claims as *'needs-review'*, and keeping corrections as procedures make curation auditable. These safeguards govern how knowledge is written, not how questions are answered: Atlas still answers confidently when the corpus lacks the information, so a verify-before-answer procedure is required.

Limitations. Deployment in a single department leaves transferability untested. The benchmark is synthetic, its gold answers are LLM-written and only partly audited, and each system ran once. Negative questions were verified as absent only against the RAG corpus, not against Atlas's larger substrate (graph, memory, and mirrored repositories), so some Atlas answers counted as fabrications may be supported there. The evaluation measures question answering, not the graph itself: schema validation guarantees well-formed nodes, not correct ones, and a stratified audit of 100 graph facts against independent sources is designed but not yet complete. Access control is enforced by user role at the portal. Propagating source-document permissions into the graph, memory, and answers is future work. Storing information about identifiable individuals also creates data-protection obligations.

Future work. We will add a verify-before-answer step that uses provenance and explicit evidence checks, complete the graph fact audit, run knowledge-graph and memory ablations, and evaluate on real user questions with independently reviewed gold answers. We also plan to package the system for other departments and to study which GDPR tasks (retention, access, correction, deletion) can be automated under human oversight.

E3 Atlas shows that a human-supervised LLM agent can build and serve a departmental knowledge base under an auditable curation methodology. It clearly outperforms an ungrounded model but not simpler grounded baselines, and its curated layers were rarely used when answering.

Acknowledgements

This research was financially supported by the Slovenian Research and Innovation Agency through the Young Researcher Grant (PR-13856), the research program Knowledge Technologies (P2-0103) and the Gravity project AI for Science (GC-0001) and European HE projects DataPact (GA 101189771) and Raido (GA 101135800). The authors acknowledge the use of Claude to support paper organization and language and grammar refinement. The authors reviewed and revised all AI-generated suggestions and take full responsibility for the content.

References

[1] Anthropic. 2025. Equipping agents for the real world with agent skills. Retrieved July 12, 2026 from https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills.

[2] Darren Edge et al. 2025. From local to global: a graph RAG approach to query-focused summarization. (2025). arXiv: 2404.16130 [cs.CL]. doi:10.48550/arXiv.2404.16130.

[3] Aidan Hogan et al. 2021. Knowledge graphs. *ACM Computing Surveys*, 54, 4, 71:1–71:37. doi:10.1145/3447772.

[4] Andrej Karpathy. 2026. LLM Wiki: a pattern for building personal knowledge bases using llms. GitHub Gist. Accessed 30 August 2026. (Apr. 2026). https://gist.github.com/karpathy/442a6bf555914893e9891c11519de94f.

[5] Patrick Lewis et al. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33, 9459–9474.

[6] Nous Research. 2026. Hermes agent: an open-source autonomous agent harness. Retrieved July 12, 2026 from https://github.com/NousResearch/hermes-agent.

[7] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2023. MemGPT: towards LLMs as operating systems. (2023). arXiv: 2310.08560 [cs.AI]. doi:10.48550/arXiv.2310.08560.

[8] Vectorize. 2026. Hindsight: long-term memory for AI agents. Retrieved July 12, 2026 from https://github.com/vectorize-io/hindsight.

[9] Lei Wang et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18, 6, 186345. doi:10.1007/s11704-024-40231-1.

[10] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

[11] Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2025. A survey on the memory mechanism of large language model-based agents. *ACM Trans. Inf. Syst.*, 43, 6, Article 155, (Sept. 2025), 47 pages. doi:10.1145/3748302.