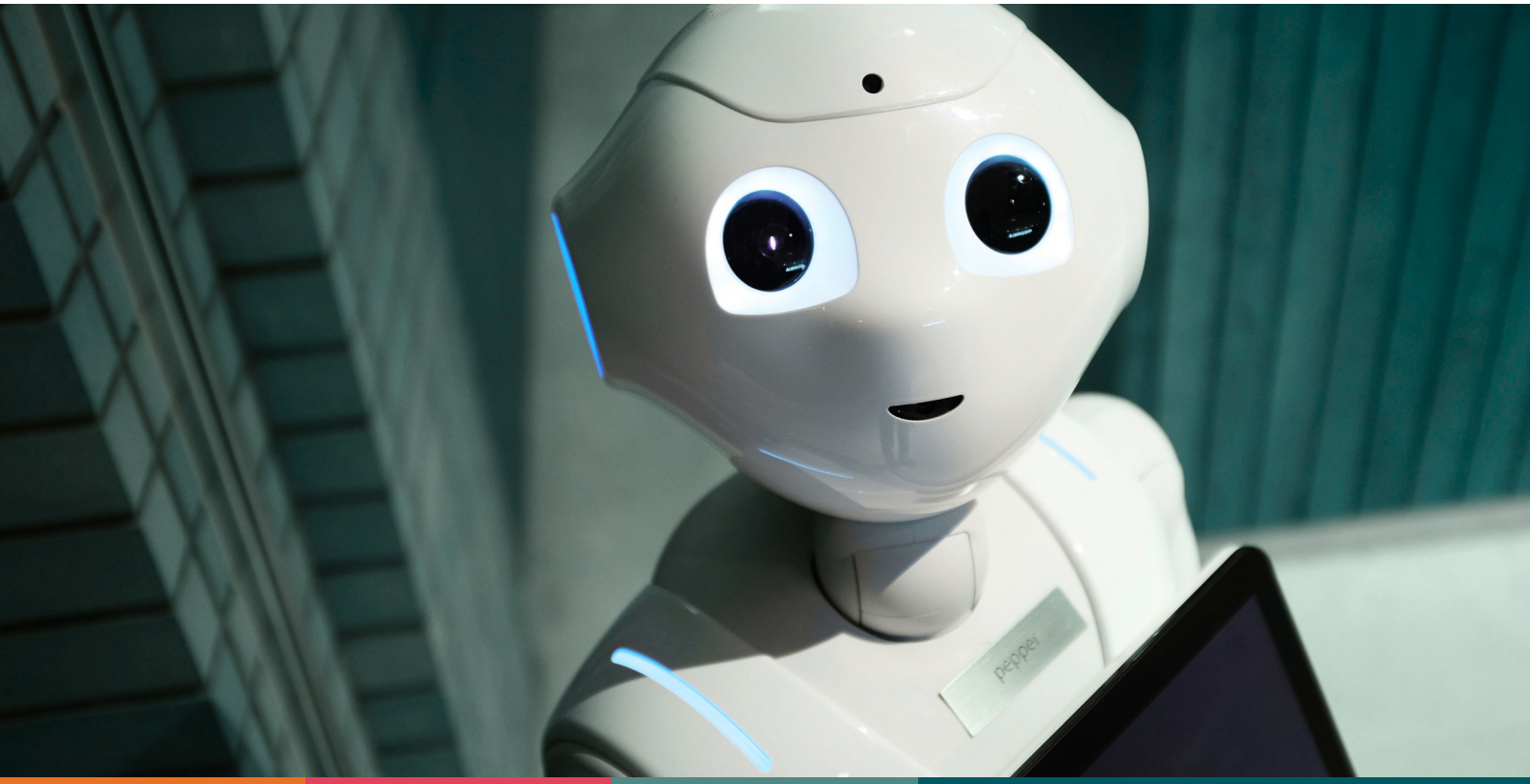




**XITASO**



Shaping the future

## **AI agents in practice**

How companies overcome fragmentation and  
make knowledge usable

Whitepaper – Artificial intelligence



# Content

|                                                               |    |                                                                          |    |
|---------------------------------------------------------------|----|--------------------------------------------------------------------------|----|
| <b>Executive Summary</b>                                      | 04 | <b>3. Quality assurance of AI agents</b>                                 | 14 |
| <b>1. AI agents for education</b>                             | 04 | 3.1. Systematic evaluation through LLM-as-Judge and regression detection | 14 |
| 1.1. Integration of heterogeneous portals through MCP servers | 04 | 3.2. Evaluation of non-deterministic systems                             | 14 |
| 1.2. Heterogeneous system landscapes                          | 05 | 3.3. Evaluation infrastructure                                           | 14 |
| 1.3. MCP servers and Agentic AI                               | 05 | 3.4. Three-dataset strategy                                              | 15 |
| 1.4. Three interaction patterns for AI agents                 | 06 | 3.5. LLM-as-Judge: Automated quality assessment                          | 16 |
| 1.5. Architecture overview                                    | 07 | 3.6. Regression detection and evaluation cycle                           | 16 |
| 1.6. Conclusion                                               | 08 | 3.7. Conclusion                                                          | 16 |
| <b>2. AI agents in customer service</b>                       | 08 | <b>4. Summary and outlook</b>                                            | 17 |
| 2.1. Intelligent product advice through RAG                   | 08 |                                                                          |    |
| 2.2. Customer service at a product manufacturer               | 09 |                                                                          |    |
| 2.3. RAG-based AI agents                                      | 09 |                                                                          |    |
| 2.4. Three areas of application for the AI agent              | 10 |                                                                          |    |
| 2.5. Architecture overview                                    | 12 |                                                                          |    |
| 2.6. Conclusion                                               | 13 |                                                                          |    |



# Executive Summary

Today, companies face two parallel challenges: Their system landscapes have grown historically and are fragmented, and the use of AI agents to overcome this fragmentation requires new approaches to quality assurance. Traditional integration strategies reach their limits when users have to switch between portals, databases, and document collections to complete a single task. AI agents offer a new way forward: they act as an intelligent interface that connects heterogeneous systems and provides users with uniform, natural language access.

This white paper highlights three key facets of this approach:

- Integration of heterogeneous systems:** Using the example of a public educational institution, the first section shows how AI agents use the Model Context Protocol (MCP) as an integration layer between educational portals and implement three interaction patterns—chat, co-editing, and delegation.

- Knowledge-based customer service:** Using the example of a medium-sized tool manufacturer, the second section demonstrates how an RAG-based AI agent combines fragmented product knowledge from PIM systems, FAQ databases, and manuals in a natural language interface, covering product advice, technical support, and intelligent escalation.
- Systematic quality assurance:** The third section presents an evaluation pipeline that uses LLM-as-Judge evaluation, three complementary data sets, and a closed regression loop to continuously and transparently ensure the response quality of an AI agent.

Each section can be read independently, but together they provide a complete picture: from architectural decisions to productive deployment to systematic quality assurance.

## 1. AI agents for education

### 1.1. Integration of heterogeneous portals through MCP servers

The integration of heterogeneous systems poses growing challenges for companies. Users have to switch between different portals and applications to complete their daily tasks, a productivity killer that is further exacerbated by the fragmentation of software landscapes.

Agentic AI and the Model Context Protocol (MCP) offer an innovative solution: AI agents act as an intelligent integration layer that coordinates specialized tools and provides users with a unified, natural language interface to their systems.

Using the example of a public educational institution, this section shows how AI agents can be successfully used to integrate educational portals. The project demonstrates the potential of this technology for companies in all industries facing similar integration challenges.

## 1.2. Heterogeneous system landscapes

Modern companies and organizations typically operate a variety of specialized systems that have grown historically and are not technically connected to each other. Users are confronted with the following problems:

- System changes:** For related tasks, multiple applications must be opened and information transferred manually.
- Knowledge loss:** Information is distributed across systems and difficult to find.
- Time consumption:** Manual research and consolidation of data costs valuable working time.
- Inconsistencies:** Without integration, data inconsistencies between systems can easily arise.

### Concrete example: Education portals

Among other things, the institute operates the following platforms for the German education sector:

| Portal              | Function                                                   |
|---------------------|------------------------------------------------------------|
| Curriculum database | Curriculum guidelines for all federal states               |
| Media library       | Educational media: videos, worksheets, interactive content |

Teachers who want to prepare a lesson currently have to:

- Research the relevant competencies in the curriculum
- Educational media: videos, worksheets, interactive content

This fragmented approach takes time and makes it difficult to create high-quality lesson plans.

## 1.3. MCP Server and Agentic AI

### What is the Model Context Protocol?

The Model Context Protocol (MCP) is an open standard that standardizes communication between AI models and external data sources or tools. Developed by Anthropic and published as an open-source standard [1], MCP enables:

- Modular tool integration:** Each system is connected via its own MCP server
- Standardized interfaces:** Uniform communication regardless of the target system
- Flexible expandability:** New systems can be added without changing the core architecture

### How an AI agent works

- Understanding the request:** The user asks a question in natural language
- Select tools:** The agent decides which MCP servers are relevant for the query
- Call up tools:** The corresponding MCP servers are addressed with structured queries
- Combine results:** The agent combines the individual results into a suitable answer .
- Execute action:** If necessary, actions such as data entries or exports are performed



## 1.4. Three interaction patterns for AI agents

The project tested three basic interaction patterns:

| Pattern    | Description                           | Example                             |
|------------|---------------------------------------|-------------------------------------|
| Chat       | Natural language dialogue interaction | Ask questions, retrieve information |
| Co-Editing | Humans and AI edit together           | Lesson planning with bidirectional  |
| Delegation | Complete handover of a task           | Automated document transformation   |

### Chat: Natural language system queries

The classic chatbot pattern enables natural language access to heterogeneous data sources.

Data sources:

**Teacher:** "What do I need to teach about photosynthesis in 7th grade biology?"

**Agent:** Searches curriculum database → Displays relevant skills and learning objectives

**Teacher:** "Find me some suitable videos on this topic."

**Agent:** Searches media library → Provides curated media list with preview images

**Characteristics:** The user asks questions, the agent answers. The interaction is sequential and query-based. The agent automatically selects the right tools and combines information from different sources.

### Co-Editing: Joint editing of an artifact

The **Lesson Planner** demonstrates an advanced interaction pattern: humans and AI work equally on a shared document.

The teacher specifies a topic, such as "Photosynthesis, Grade 7, Bavaria." The agent independently researches curriculum references and suitable media and fills in the lesson plan. The teacher can intervene manually at any time and make adjustments. Additional requests, such as "Add another quiz," are implemented by the agent in a context-sensitive manner.

**Core functions:**

- Bidirectional interaction:** Both the AI and the teacher can edit the plan.
- Context awareness:** The AI takes manual

changes into account when making further suggestions

- Structured output:** Learning objectives, activities, and materials are clearly presented

**Export to learning management systems:** The lesson plans created can be exported directly as **Common Cartridge** and imported into Moodle, itslearning, or other LMS.

**Characteristics:** Both sides, humans and AI, can modify the shared artifact at any time. The agent acts as a co-author, making suggestions and taking on tasks, while the teacher retains full control.

### Delegation: Complete task transfer

The third pattern goes one step further: the user transfers a complete task to the agent and receives the finished result.

**Content Transformer:** One example is automated document transformation. A teacher uploads an existing document (e.g., a worksheet), and the agent:

- analyzes the content and structure
- Enriches the document with curricular references
- Selects suitable media from the media

library

**Characteristic:** The user defines the goal, and the agent carries out all steps independently. Interaction is limited to input and result transfer, ideal for repetitive or time-consuming transformation tasks.

## 1.5. Architecture overview

The chosen **modular architecture** clearly separates responsibilities and uses standardized interfaces for communication between components. Each agent is optimized for its appli-

cation, with specific prompts, workflows, and tool selection.

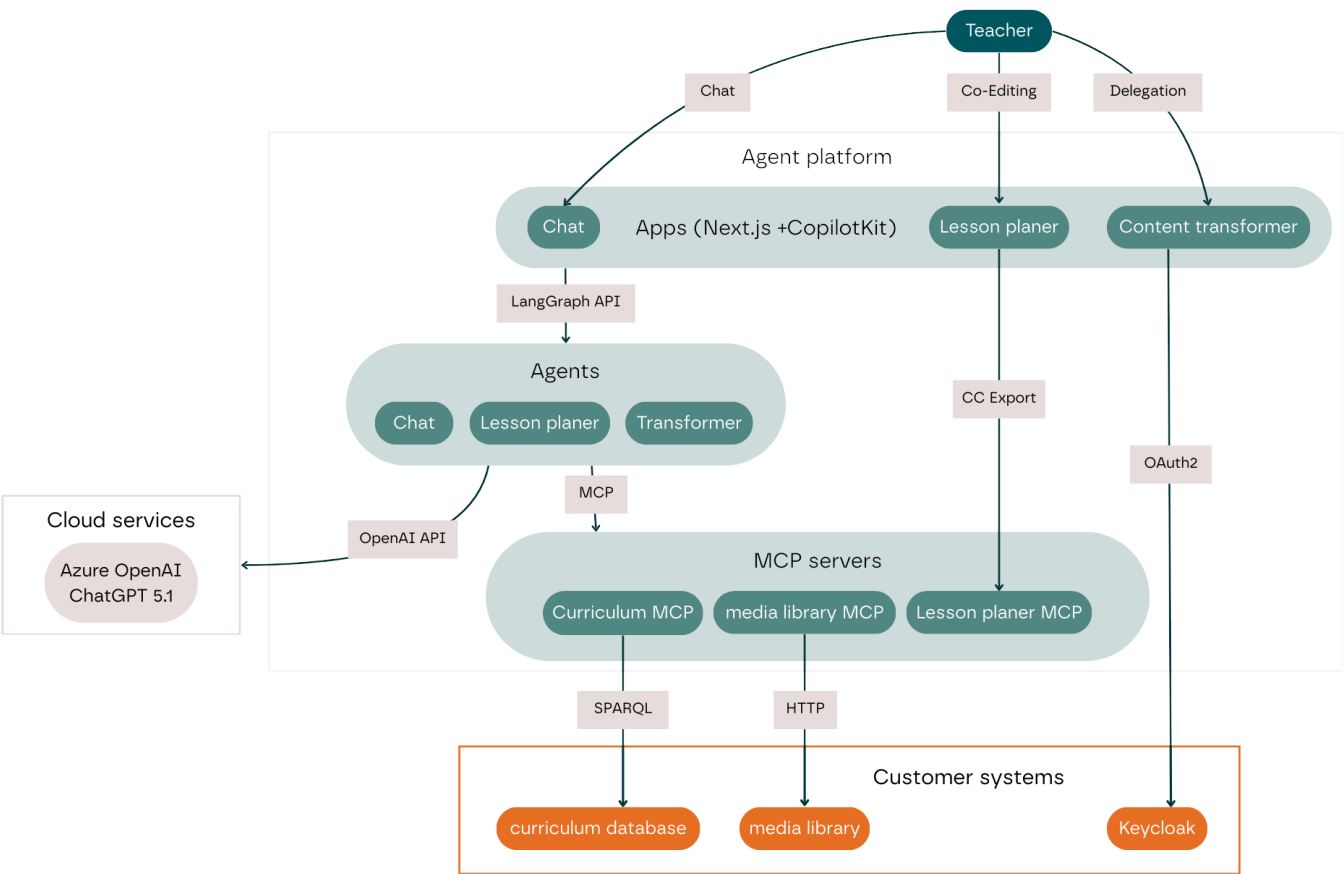


Fig. 1: Architecture of the educational institute



| Aspect           | Advantage                                                                       |
|------------------|---------------------------------------------------------------------------------|
| Modularity       | MCP servers can be developed and updated independently                          |
| Extensibility    | Integrate new systems via MCP servers, new applications via agents              |
| Maintainability  | Changes to target systems only require adjustments to the respective MCP server |
| Scalability      | Each component can be scaled independently                                      |
| Data sovereignty | Data remains in the source systems; no central data storage                     |

## 1.6. Conclusion

The project demonstrates how Agentic AI and the Model Context Protocol can effectively integrate heterogeneous system landscapes. AI agents act as an intelligent connection layer, that accesses existing systems via MCP servers without having to modify them.

The three proven interaction patterns—chat, co-editing, and delegation—cover a wide range of use cases: from simple information retrieval to the collaborative creation of complex artifacts to fully automated task processing.

The modular architecture makes it possible to integrate both new data sources via MCP servers and new applications via additional agents. This approach can be applied far beyond the education sector to any organization with heterogeneous system landscapes.

While MCP servers solve integration at the system level—the agent uses tools that access existing portals—many companies face a different challenge: it is not the connection between systems that is lacking, but rather intelligent access to deep domain knowledge scattered across databases, documents, and FAQ collections. Retrieval-Augmented Generation (RAG) offers the right approach for this use case, as the following example from a product manufacturer's customer service department shows.

# 2. AI agents in customer service

## 2.1. Intelligent product advice through retrieval-augmented generation

The manufacturer is a company in the power tools and garden equipment sector with several thousand products in five different product categories. The challenge: Product knowledge is scattered across PIM systems, FAQ databases, and product manuals and is difficult for customers and employees to access.

A **RAG-based AI agent** (retrieval-augmented generation) combines these knowledge

sources in a single, natural language interface. The agent masters three central areas of responsibility: **product advice**, **technical support**, and **intelligent escalation** to human customer service. It is used in two modes: as an internal tool for employees and as an external chatbot for end customers in the web shop.

## 2.2. Customer service for a product manufacturer

A manufacturer with several thousand products in five different product groups faces a fundamental problem: providing expert advice requires **enormous product knowledge**, which in practice is spread across multiple systems.

Three pain points characterize everyday life:

- **Fragmented knowledge:** Product specifications are stored in the PIM system, troubleshooting instructions in the FAQ database, and assembly and maintenance instructions in product manuals. None of these sources alone provides a complete picture.
- **Time-consuming training:** New customer service employees must learn which information can be found in which system. The result: long onboarding times and inconsistent answers.

- **Scaling support:** Customers expect fast, accurate answers, but human support does not scale with growing inquiry volumes.

**A concrete example:** A customer asks, "Which lawn mower is suitable for my 100 m² garden?" To answer this seemingly simple question, an employee must weigh up the cutting width, motor power, and device type of various models, check availability in the store, identify suitable accessories, and, in the case of battery-powered devices, clarify the compatibility of the battery series. This requires mentally comparing several systems. It is a time-consuming process that becomes even more complex for more complex questions.

| Source          | Content                                                                                   |
|-----------------|-------------------------------------------------------------------------------------------|
| PIM system      | Product data:<br>specifications, prices, availability, accessories, battery compatibility |
| FAQ database    | Questions and answers on troubleshooting, usage, and common problems                      |
| Product manuals | 100+ documents:<br>assembly instructions, maintenance instructions, safety information    |

## 2.3. RAG-based AI agents

### What is retrieval-augmented generation?

**RAG** combines the language capabilities of large language models (LLMs) with the targeted retrieval of external knowledge sources. The basic principle: The language model "knows" nothing about the manufacturer's products. Instead, it retrieves **real data from vector databases** for each query and uses it to generate an informed response.

This approach solves two fundamental problems at once. A **pure language model** without external data tends to hallucinate: it generates plausible-sounding but factually incorrect answers [3]. A **pure search engine**, on the other hand, delivers correct hits but cannot summarize them into a coherent, context-related answer. RAG combines the best of both worlds: **fact-based answers in natural language** [2].



## Hybrid search: the best of both worlds

To retrieve relevant information, the agent relies on a **hybrid search strategy** that combines two methods:

- **Semantic search** (dense embeddings, Jina AI): Understands the meaning behind a query. The question "device for sawing wood" also finds band saws and table saws, even though none of these words appear in the query.

### How the AI agent works

The agent works according to the **ReAct pattern** (Reasoning + Acting) [4]: It thinks about the query, selects appropriate tools, evaluates the results, and decides whether further steps are necessary. It has **15 specialized tools** at its disposal, from product search and FAQ queries to manual research.

The process follows a structured procedure:

1. **Understanding the query:** The agent analyzes the customer's question and identifies the information required.
2. **Select tools:** Based on the analysis, the agent selects the appropriate tools: product search, FAQ database, manual research, or availability check

- **Keyword search** (BM42): Delivers precise hits for specific terms such as item numbers or model names such as "Model A."

The **combination of both methods** achieves a higher hit rate than either method alone: semantic understanding for open-ended questions, keyword precision for specific product queries.

3. **Retrieve knowledge:** The selected tools search the respective vector databases and deliver relevant results
4. **Synthesize the answer:** The agent combines the results into a coherent, German-language answer
5. **Escalate if necessary:** If the agent realizes that it cannot help, it forwards the request to human customer service

## 2.4. Three areas of application for AI agents

### Product advice: From the question to the right product

The most common use case is needs-based product advice. Customers describe their use case, and the agent translates this into specific product recommendations.

#### Sample dialogue:

**Customer:** "Which lawn mower do you recommend for a 100 m² garden?"

**Agent:** searches product database with filters for garden tools and lawn mowers → recommends three suitable models with cutting width, motor power, and price

**Customer:** "What is the difference between model A and model B?"

**Agent:** uses comparison tool → creates a comparison of both models with all relevant specifications

The agent actively **draws conclusions:** Based on the garden size, they determine which cutting width and motor power are appropriate, check availability in the shop, and can clarify compatibility with existing batteries for battery-powered models. The manufacturer offers three incompatible battery series. This is a common stumbling block, which the agent avoids using a specialized tool to check battery compatibility.

**Characteristics:** The agent combines semantic search with structured product filters and provides personalized recommendations based on the customer's use case.

## Technical support: Knowledge from manuals and FAQs

The second area of application is **technical support**, from quick troubleshooting tips to detailed maintenance instructions.

#### Sample dialogue:

**Customer:** "Why is white smoke coming out of my gas lawn mower?"

**Agent:** searches FAQ database → finds troubleshooting entry → provides step-by-step diagnosis: check oil level, check air filter, clean carburetor if necessary

**Customer:** "How do I store the device in winter?"

**Agent:** searches product manual → finds section on winter storage → provides specific instructions: drain fuel, check spark plug, store in a dry place

The agent searches **two knowledge sources in parallel:** the FAQ database for frequently asked questions with concise answers and the product manuals for detailed, product-specific instructions. The item number can be used to search specifically in the manual for a particular product or across all available manuals.

**Characteristics:** The agent searches both the FAQ database and product manuals and provides context-related instructions, from quick troubleshooting tips to detailed maintenance instructions.

### Intelligent escalation: Seamless transfer to customer service

An AI agent that **knows its limits** inspires more confidence than one that tries to answer every question. The third area of application is therefore **intelligent escalation:** targeted transfer to human customer service when the agent cannot help.

Technically, escalation is implemented as a **separate tool** that the agent can call up independently. The language model decides when a transfer is appropriate based on its understanding of the context and selects a suitable **escalation category:**

| Category           | Example                                          | Action                   |
|--------------------|--------------------------------------------------|--------------------------|
| Warranty extension | "Register for warranty"                          | Registration form        |
| Order questions    | "Where is my delivery?"                          | Order status form        |
| Complaint          | "Product arrived defective"                      | Complaint form           |
| Return             | "I would like to return the device"              | Return form              |
| Explicit request   | "I would like to speak to a representative"      | General customer service |
| Low confidence     | Agent cannot find an answer after using the tool | General customer service |



This approach has a decisive advantage over rule-based systems: the agent recognizes the need for escalation **based on context**. Instead of relying on rigid keyword lists or fixed thresholds, they understand the intention behind a request. The statement "The device no longer works, I only bought it two weeks ago" is correctly recognized as a complaint and forwarded to the appropriate form.

## 2.5. Architecture overview

The architecture follows a clear layered principle: an API layer receives requests, the RAG pipeline coordinates the process, the AI agent performs reasoning and accesses the knowledge sources via specialized tools.

Each tool-call searches the respective vector

**The key point for management is** that the agent does not attempt to handle warranty claims, order status inquiries, or returns themselves. They recognize these cases and forward **them to the correct human process**, reliably and without friction.

database using hybrid search, semantically via Jina AI Embeddings and keyword-based via BM42 and returns the most relevant results to the agent.

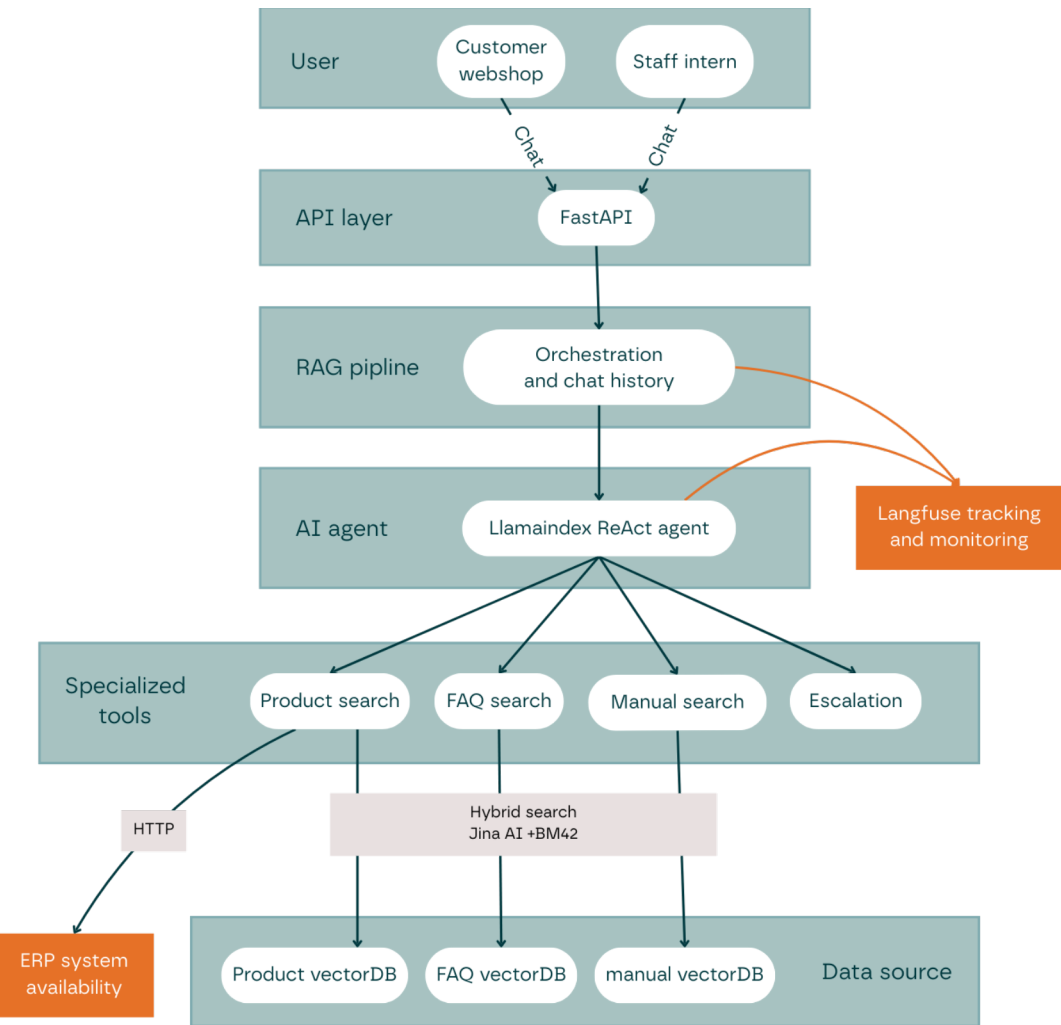


Fig. 2: Architecture of the tool manufacturer

## Two modes of operation

The system serves two target groups via different configurations from a single code base:

| Aspect       | Internal (employees)                         | External (customer)                  |
|--------------|----------------------------------------------|--------------------------------------|
| Target group | Customer service employees                   | End customers in the web shop        |
| Inventory    | Exact quantities from the ERP system         | Only "available" / "unavailable"     |
| Escalation   | Not available (employee is customer service) | Transfer to human customer service   |
| Tools        | 14 specialized tools                         | 15 tools (including escalation tool) |

## Advantages of this architecture

| Aspect        | Advantage                                                               |
|---------------|-------------------------------------------------------------------------|
| Up-to-date    | Product data is retrieved from source systems, not stored in the model  |
| Extensibility | New knowledge sources can be integrated via additional vector databases |
| Traceability  | Every answer is based on referenceable source documents                 |
| Quality       | Consistent tracing via Langfuse for monitoring and evaluation           |

## 2.6. Conclusion

The RAG-based AI agent shows how **fragmented product knowledge**, distributed across PIM systems, FAQ databases, and product manuals, can be combined in a single, natural language interface.

The three areas of application – **product advice, technical support and intelligent escalation** – cover the central requirements of a product manufacturer's customer service. The agent knows its limits: using its own escalation tool, it recognizes when a request requires human judgment and reliably forwards it to customer service.

With two modes of operation, internal for employees and external for customers, a single system serves different target groups with customized information.

Deploying a RAG agent productively is one challenge – systematically ensuring the quality of its responses over time is another. Since AI agents generate different but equally correct responses to identical inputs, traditional software testing falls short. The following evaluation methodology was developed to close this gap.



### 3. Quality assurance of AI agents

#### 3.1. Systematic evaluation through LLM-as-Judge and regression detection

A RAG-based AI agent generates different but equally correct answers for identical inputs. Classic software tests that check exact outputs fall short here. Instead, open response formats, multi-stage processing, and context dependency require semantic judgment: it is not the exact wording that counts, but whether a response is correct, complete, and relevant in terms of content. The solution lies in a combination of automated LLM evaluation, structured data sets, and continuous regression detection.

#### 3.2. Evaluation of non-deterministic systems

A RAG-based AI agent generates different but equally correct answers for identical inputs. Classic software tests that check exact outputs fall short here. Instead, open response formats, multi-stage processing, and context dependency require **semantic judgment**: it is not the exact wording that counts, but whether a response is correct, complete, and relevant in terms of content. The solution lies in a combination of automated LLM evaluation, structured data sets, and continuous regression detection.

#### 3.3. Evaluation Infrastructure

The evaluation pipeline relies on six specialized metrics based on the DeepEval framework [10], with central concepts such as faithfulness and answer relevance building on the RAGAS framework [9]:

| Metric             | Checks                                                 | Type            |
|--------------------|--------------------------------------------------------|-----------------|
| Correctness        | Factual accuracy of the answer                         | Reference-based |
| Completeness       | Completeness in relation to the expected answer        | Reference-based |
| Answer Relevancy   | Relevance to the question asked                        | Reference-free  |
| Fluency            | Linguistic quality (German)                            | Reference-free  |
| Faithfulness       | Accuracy of facts in relation to the sources consulted | Reference-free  |
| Chat History Usage | Use of chat history in multi-turn dialogues            | Reference-free  |

This selection follows the **5-metric rule**: A few targeted metrics provide a clearer signal than a large number of redundant evaluations [10]. The combination of reference-based and reference-free metrics enables both comparison with expected responses and evaluation of aspects that do not require a reference response.

**Langfuse** [8] serves as the central platform for experiments and tracing. An asynchronous

evaluation engine calls the RAG pipeline for each data set element and evaluates the response against all six metrics. Mistral is deliberately used as **the evaluation model**, a different model from the generation model. This separation prevents the proven tendency of language models to systematically evaluate their own outputs better [5][6][11].

#### 3.4. Three-dataset strategy

A single test collection cannot be an integration test, coverage test, and regression test at the same time. Therefore, the evaluation relies on **three complementary data sets**:

| Data set                 | Scope          | Quality standards                 | Checks                                               |
|--------------------------|----------------|-----------------------------------|------------------------------------------------------|
| Functional integration   | ~30 elements   | Mixed, including borderline cases | Are all data sources and tools functioning properly? |
| Question type coverage   | ~35 elements   | Mixed, including borderline cases | Are all user question types covered?                 |
| Regression gold standard | 60-80 elements | All human-validated (0.9-1.0)     | Has something gotten worse?                          |

The **integration data set** checks whether all system components are working together correctly (PIM, ERP, FAQ, manuals, escalation, chat history) with approximately five elements per source and deliberately includes borderline cases. The **coverage data set** tests all user question types (product recommendation, comparison, purchase advice, support, returns, delivery) and follows the recommended distribution of 60-70% typical queries, 15-20% borderline cases, and 10-15% adversarial

inputs [11]. The **gold standard** contains 60-80 question-answer pairs where the model's response has been validated by subject matter experts as very good or perfect ( ), distributed across all data sources and question types [5][7].

The three datasets complement each other: while the gold standard checks that good answers remain good, the coverage datasets reveal new error modes and edge cases.



### 3.5. LLM-as-Judge: Automated Quality Assessment

The evaluation follows a **three-stage assessment hierarchy** [5][6][7]: Where possible, deterministic, code-based checks are used (stage 1). For semantic quality dimensions such as correctness and completeness, LLM-based evaluation takes over (stage 2). Human evaluation (level 3) is reserved for periodic calibration and gold standard validation.

Four design decisions ensure the reliability of automated evaluation:

1. **Model separation:** The evaluation model (Mistral) is distinct from the generation model to eliminate systematic self-bias [5][6][11].

2. **Categorical scales:** Discrete rather than continuous evaluation scales, as language models evaluate more inconsistently on fine-grained scales [10][11].
3. **Detailed criteria:** Each metric defines specific criteria. The evaluation model first formulates a justification before giving an evaluation [5].
4. **Periodic calibration:** Subject matter experts regularly review a sample of automated assessments with the goal of achieving 80-90% agreement [8].

### 3.6. Regression detection and evaluation cycle

Quality declines are detected on three levels: **Absolute thresholds** define minimum values per metric (e.g., no gold standard element below 0.7, overall average maximum 5% below baseline). **A relative comparison** contrasts two evaluation runs, calculates deltas, and visualizes changes. **Element tracking** tracks which specific question-answer pairs have changed between versions and identifies patterns in the regressions.

The data sets grow continuously through a **closed loop** [7][12]: Errors are included in the coverage data sets as new test cases. After correction and human validation, confirmed corrections are added to the gold standard. In this way, test coverage grows organically with the system.

### 3.7. Conclusion

Quality assurance for an RAG-based AI agent requires a fundamentally different approach than traditional software testing. LLM-based evaluation systems take on the role of human reviewers and check for correctness, completeness, and relevance according to defined evaluation criteria. The three-dataset strategy

neatly separates functional integration, question type coverage, and regression detection. Together with the evaluation cycle, which systematically converts production errors into new test cases, this creates a system that continuously improves itself.

## 4. Conclusion and outlook

This white paper has highlighted three key aspects of the practical application of AI agents: **MCP-based integration** of heterogeneous systems, **RAG-supported knowledge access** in customer service, and **systematic quality assurance** through LLM-as-Judge and regression detection. Together, they form a comprehensive approach — from architectural decisions to productive operation to continuous quality control.

Three insights are transferable across industries:

- **Augment rather than replace:** AI agents do not replace existing systems or human expertise. They create an intelligent layer on top that improves access, usage, and quality. In education, they connect portals, in customer service they combine knowledge sources, and in evaluation they take on the role of human reviewers for scalable assessments.
- **Architecture follows the challenge:** MCP is suitable for integrating heterogeneous systems with standardized interfaces. RAG is the right approach when deep domain knowledge from unstructured sources needs to be made accessible. The choice of architecture depends on the specific problem, not on the technology.

- **Evaluation is not an option, but a prerequisite:** Non-deterministic systems require non-deterministic testing methods. Anyone who operates an AI agent productively without systematically measuring its response quality is flying blind. The closed evaluation cycle — from automated assessment to regression detection to gold standard validation — is an integral part of any production-ready system.

The approaches presented are not limited to the domains described. Any organization dealing with fragmented system landscapes, distributed expertise, or the quality assurance of AI-supported processes can benefit from the patterns described here.

Looking ahead, further developments are on the horizon: The maturation of MCP as a cross-industry standard will further simplify the interoperability of AI agents with existing systems. Multi-agent systems, in which specialized agents solve complex tasks by dividing the work, will open up new use cases. And the further development of LLM-as-Judge methods will further narrow the gap between automated and human evaluation.



# Your Contact

We want to work with you to make your AI project a success.  
Get in touch!



## Dr. Simon Stieber

Senior AI Engineer

Phone +49 89 69 31 470-110

Email [simon.stieber@xitaso.com](mailto:simon.stieber@xitaso.com)



## Matthias Widmann

Sales & Business Development

Phone +49 821 885882-228

Email [matthias.widmann@xitaso.com](mailto:matthias.widmann@xitaso.com)



## About XITASO

Your partner for high-end software engineering in the age of AI

As a digitalization partner and expert in high-end software engineering, XITASO works with its customers on demanding digitalization projects to create customized solutions that scale and function securely from the hardware to intuitive user interfaces and into the cloud.

The company supports B2B customers in identifying digitalization potential and optimizing business processes. With industry experience, enthusiasm for innovative technologies such as artificial intelligence, and a passion for excellent collaboration, XITASO helps its customers master the challenges of digital transformation and secure their market advantage.

2011

Founded

13

Locations

240+

Employees  
(12/2024)

22

Revenue 2024  
in mio. €



## Sources

[1] Anthropic, “Introducing the Model Context Protocol,” 2024. <https://www.anthropic.com/news/model-context-protocol>

[2] P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” NeurIPS 2020. <https://arxiv.org/abs/2005.11401>

[3] Z. Huang et al., “A Survey on Hallucination in Large Language Models,” 2023. <https://arxiv.org/abs/2311.05232>

[4] S. Yao et al., “ReAct: Synergizing Reasoning and Acting in Language Models,” ICLR 2023. <https://arxiv.org/abs/2210.03629>

[5] Anthropic, “Create Strong Empirical Evaluations,” 2025. <https://docs.anthropic.com/en/docs/build-with-claude/develop-tests>

[6] OpenAI, “Getting Started with OpenAI Evals,” 2024. [https://cookbook.openai.com/examples/evaluation/getting\\_started\\_with\\_o-penai\\_evals](https://cookbook.openai.com/examples/evaluation/getting_started_with_openai_evals)

[7] LangSmith, “Evaluation Concepts,” 2025. <https://docs.smith.langchain.com/evaluation>

[8] Langfuse, “Model-Based Evals,” 2025. <https://langfuse.com/docs/scores/model-based-evals>

[9] RAGAS, “Metrics Documentation,” 2025. <https://docs.ragas.io/en/stable/concepts/metrics/>

[10] Confident AI, “LLM Evaluation Metrics,” 2025. <https://www.confident-ai.com/blog/llm-evaluation-metrics-everything-you-need-for-llm-evaluation>

[11] Arize AI, “LLM Evaluation: The Definitive Guide,” 2025. <https://arize.com/blog-course/llm-evaluation-the-definitive-guide/>

[12] H. Husain, “Your AI Product Needs Evals,” 2024. <https://hamel.dev/blog/posts/evals/>

## List of figures

|                    |                                     |    |
|--------------------|-------------------------------------|----|
| • <b>Figure 1:</b> | Architektur des Bildungsinstituts   | 07 |
| • <b>Figure 2:</b> | Architektur des Werkzeugherstellers | 12 |

**A big thank you to everyone involved!**

Authors:  
Dr. Simon Stieber  
Simon Adameit  
Anton Winschel

enjoy  
digital  
excellence



Learn more about  
AI at XITASO

**XITASO GmbH**  
**IT & Software Solutions**  
Austraße 35 · D-86153 Augsburg

**Phone** +49 821 885882-

**Email** [info@xitaso.](mailto:info@xitaso.)

**Web** [www.xitaso.com](http://www.xitaso.com)