



Article

LLM-Based Knowledge Engineering for DSS Collaborative Knowledge Bases: Approach and Pilot Study

Igor Glukhikh, Kirill Glukhikh and Dmitry Glukhikh *

School of Computer Science, University of Tyumen, 6 Volodarskogo St., Tyumen 625003, Russia

* Correspondence: gluhidmitry@gmail.com

Abstract

The creation of collaborative knowledge bases for decision support systems (DSS) mitigates the subjectivity of individual experts and enhances overall system efficacy. However, traditional knowledge engineering approaches are highly labor-intensive when eliciting and integrating expert knowledge, as they require extensive, expert-level interaction between knowledge engineers and domain specialists. Modern large language models (LLMs) and retrieval-augmented generation (RAG) technologies present novel opportunities for overcoming these limitations. This study presents a pilot investigation to assess the potential of LLM-based knowledge engineering for developing collaborative knowledge bases within knowledge-based DSS, thereby assisting decision-making in complex operational scenarios involving technical systems. The article proposes an LLM-based approach for creating collaborative knowledge bases, including extraction, consolidation of expert knowledge and evaluation of their operability. To implement and evaluate the proposed approach, specialized prompts were engineered, and pilot experiments were conducted to generate consolidated knowledge cases through expert-LLM interactions. The resulting knowledge cases were subsequently applied in an experimental decision-making inference procedure for fault diagnosis in gas-fired heating boilers. During this inference process, an LLM agent, guided by tailored prompts and a RAG-enabled knowledge base, interactively queries the user to identify the specific issue and subsequently proposes a contextually appropriate solution. Throughout this study, the LLMs demonstrated capabilities in dialogue management, expert knowledge elicitation, and knowledge consolidation, successfully facilitating the creation of a collaborative knowledge base grounded in the “Event-Cause-Symptoms-Action” model. The findings highlight the viability of future research in LLM-based knowledge engineering and support the further advancement of the “LLM-as-knowledge-engineer” paradigm.

Academic Editor: Mehmed
Kantardzic

Received: 12 May 2026

Revised: 2 July 2026

Accepted: 3 July 2026

Published: 5 July 2026

Copyright: © 2026 by the authors.
Licensee MDPI, Basel, Switzerland.
This article is an open access article
distributed under the terms and
conditions of the [Creative Commons
Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Keywords: collaborative knowledge base; knowledge-based decision support system; large language model (LLM); retrieval-augmented generation (RAG); case-based reasoning (CBR); LLM-based knowledge engineering

1. Introduction

Relevance of the Study

The rapid integration of large language models (LLMs) into various spheres of society offers new opportunities for developing technologies that facilitate human interaction. Currently, the most common application of LLMs is responding to user queries. However, numerous studies indicate that LLMs are also highly effective in more complex interactions and tasks. For instance, LLMs have demonstrated their capability to enhance human cognitive abilities and support researchers in scientific activities [1–3], decision-making processes [4,5], and project and production management [6,7]. Furthermore, LLMs can provide novel methodologies for solving well-established problems. A classic example is image recognition, which traditionally involves determining the class to which an image or an object within it belongs.

Traditionally, this task has been addressed using neural network models trained on large datasets of pre-categorized images. However, by combining computer vision models with LLMs, it is now possible to recognize objects by matching images with textual descriptions. This approach eliminates the need to create datasets with predefined classes. Another emerging opportunity in this domain of computer vision includes tasks such as scene description, image captioning, and visual question answering [8].

With the advent of LLMs incorporating retrieval-augmented generation (RAG) technology, new possibilities have emerged in the well-established field of artificial intelligence, particularly in the development of knowledge-based decision support systems (KB DSS) [9–12]. As highlighted in a recent roadmap on post-LLM AI models, the integration of knowledge empowerment and collaborative, co-evolving architectures opens transformative directions for such systems [13]. KB DSS are designed to solve complex, practically significant problems by leveraging the expertise of domain specialists, which is formalized within DSS knowledge bases using specialized knowledge representation models.

A promising direction for KB DSS is the creation and utilization of collaborative knowledge bases. The advantages and capabilities of such systems have been extensively discussed in the context of traditional knowledge representation methods and ontological engineering [14–16]. In these knowledge bases, the insights of diverse individuals complement one another, making them powerful tools for addressing complex industrial or societal challenges.

This approach helps overcome the subjectivity inherent in decisions proposed by individual specialists. Furthermore, collaborative knowledge engineering enables the creation of knowledge bases without relying exclusively on highly sought-after domain experts, who are often in short supply and have limited availability.

While a collaborative knowledge base can significantly enhance a KB DSS's competence in solving applied problems, the complexity of creating and maintaining such systems using traditional knowledge engineering approaches poses a major obstacle to their widespread adoption.

Eliciting expert knowledge and formally representing it for use in decision-making algorithms remains a key knowledge engineering challenge in DSS development. This challenge becomes even more pronounced in collaborative knowledge bases, which require extracting and consolidating knowledge from multiple sources. Specifically, the following difficulties require particular attention:

- Potential inconsistencies or discrepancies in the knowledge provided by the experts and specialists involved in building the collaborative knowledge base, necessitating additional efforts to align and consolidate this information.

- Increased labor intensity and time costs associated with extracting and integrating expert knowledge from diverse sources, which consequently raises the overall cost of knowledge base construction.

We hypothesize that LLM-based technologies offer a new suite of tools to address these challenges in DSS knowledge base development. On the one hand, LLMs have already demonstrated their capabilities in human–AI interaction and process automation across various applied tasks. On the other hand, RAG technology has shown significant potential for developing question-answering systems grounded in domain-specific knowledge bases. This provides strong grounds to expect that LLM/RAG systems will play an active and effective role in eliciting expert knowledge and constructing DSS knowledge bases. Recent research further supports this trajectory by demonstrating that formal knowledge-augmented language models can provide the explainable and robust reasoning capabilities essential for knowledge engineering tasks [17].

This work aims to conduct a pilot study to assess the potential of LLM-based knowledge engineering for creating collaborative knowledge bases within KB DSS, thereby assisting decision-making in complex operational scenarios involving technical systems.

To achieve this, we:

- Developed and described a general LLM-based approach for creating collaborative knowledge bases, encompassing knowledge elicitation, consolidation, and functional assessment;
- Proposed a structured knowledge representation model based on cases, which facilitates knowledge elicitation, consolidation, and application in decision reasoning. This enables the integration of RAG and case-based reasoning (CBR) [18] for rigorous decision-making grounded in structured knowledge;
- Developed and experimentally tested a prompt engineering framework to implement the proposed approach within a technical case study;
- Developed a methodology and conducted an evaluation of the prepared knowledge cases for decision reasoning in scenarios where ground-truth reference solutions were unavailable;
- Analyzed the results and limitations of the proposed approach, outlining future directions for LLM-based knowledge engineering in KB DSS development.

The remainder of this paper is structured as follows. Section 2 reviews recent research on the application of LLMs in knowledge engineering, RAG, and CBR that informed this study. Section 3 provides a brief overview of CBR and RAG approaches, demonstrating the potential to enhance RAG capabilities using knowledge structuring models adopted from CBR. Section 4 details the proposed LLM-based knowledge engineering approach, including the overall framework for knowledge elicitation, consolidation, and experimental evaluation, alongside the specific prompts developed for these tasks. Subsequently, Section 5 presents experimental findings on knowledge integration for cases involving complementary and contradictory information. It also details the methodology for evaluating knowledge base performance during solution inference. Finally, Sections 6 and 7 summarize the findings, address limitations, and outline future research directions.

2. Related Works

The rapid proliferation of LLMs has attracted increasing attention from researchers across various scientific disciplines. Knowledge engineering, a discipline concerned with the acquisition, representation, and application of knowledge in applied intelligent systems, has also been significantly influenced by this trend. For instance, the authors of

[19] advocated for a revision of knowledge engineering practices and methodologies in light of LLMs, demonstrated new capabilities for natural language-based knowledge engineering, and outlined several issues that must be addressed to advance the field. On the one hand, LLMs have demonstrated substantial potential for extracting knowledge directly from texts. In this context, the Named Entity Recognition (NER) task, well-established in Natural Language Processing (NLP), serves as a prime example. For instance, the authors of [7] employed LLMs for knowledge extraction and the construction of a domain taxonomy for additive manufacturing processes through named entities. Similarly, the authors of [20] utilized prompt engineering techniques to develop a standardized prompt structure specifically designed for NER.

Concurrently, knowledge-based decision support systems (KB DSS) require knowledge that encompasses not only named entities, facts, and events but also the complex dependencies among them. This enables the construction of knowledge bases in the classical sense, serving as a vehicle for simulating human reasoning during problem-solving. The authors of [21] demonstrated that LLMs, when guided by specialized instructions and prompts, can be effectively employed to construct a case base in a complex real-world domain and to develop a case-based reasoning (CBR) decision-making system.

However, a substantial portion of expert empirical knowledge remains uncaptured in published texts. Consequently, the integration of LLMs into knowledge engineering methodologies also entails their application to the elicitation of tacit knowledge directly from specialists—through interviews, evaluations, and subsequent processing of the elicited data. For instance, the authors of [22] proposed a multi-agent LLM system for acquiring knowledge about business processes, specifically by interviewing domain experts to identify processes and their characteristics. Their findings indicated that business process models constructed with LLMs can surpass the quality of those produced by junior analysts.

The present study extends this line of inquiry. LLMs use specialized prompts and algorithms and assume the role of knowledge engineers, collaborating with experts to elicit and structure their empirical knowledge. We proceed from the premise that individual empirical knowledge may lack the completeness and accuracy required for effective execution of practical tasks. To address such limitations, the authors of [14–16] proposed the creation of collaborative knowledge bases. As the number of participants increases and the processes involved in building such collaborative knowledge bases grow in complexity, the need to automate these processes becomes increasingly pressing. LLMs functioning as knowledge engineers can serve as a pivotal element in achieving this automation.

In the context of KB DSS development, not only is knowledge extraction important, but so is the creation of methods and tools for its effective application in problem analysis and solution inference. The authors of [23] investigated the applicability of LLMs for decision-making in the domain of building engineering system management. We utilize this exact domain to illustrate the application of our approach, specifically for troubleshooting gas-fired boilers in building heating systems. The relevance of these tasks is tied to ensuring the operational efficiency of engineering systems, as well as the safety and comfort of building occupants [24]. However, the results of the study by [23] demonstrated that the reasoning processes of standalone LLMs are overly simplistic and fail to provide the reliability required for autonomous decision-making.

The advent of RAG represents a crucial advancement that enhances the ability of LLMs to address such challenges by leveraging external knowledge beyond their pre-training data [25,26]. In its basic form, RAG technology involves creating a vector store of documents divided into text chunks. These chunks, acting as knowledge components, are retrieved and passed alongside user queries to the LLM to generate contextually grounded responses.

However, attempts to deploy LLM/RAG systems in industrial applications face challenges related to massive document volumes, ambiguities in semantic search results within corporate knowledge bases, and their limited suitability for decision-making in complex scenarios. Consequently, recent research has focused on augmenting RAG with logical inference methods [27–29] and knowledge representation models that feature higher levels of formalization [17,30]. This represents a significant step toward the creation of neuro-symbolic intelligence, fostering synergy between LLM neural networks and classical AI methods. Of particular interest is the augmentation of LLMs with RAG through CBR inference methods. The CBR approach has proven highly effective in DSS that require rapid solution inference for problem situations by leveraging an existing case base and making decisions based on similar past cases. Several recent studies have demonstrated the potential of CBR to enhance LLM effectiveness and overcome the limitations of naïve RAG in searching large knowledge volumes [31], inferring solutions [32,33], and performing crucial tasks such as justifying the inferred solutions [34].

Building upon these studies, we develop an LLM-based collaborative knowledge engineering approach and demonstrate its potential for decision-making.

3. Materials and Methods

In this section, we describe the core principles of CBR and RAG. Although these approaches emerged at different stages of artificial intelligence development, they share fundamental similarities. In particular, both rely on similarity analysis and the retrieval of relevant objects from an existing knowledge base. However, CBR supports a broader range of object representation methods and, consequently, a wider variety of similarity metrics. RAG, in turn, operates on textual data, employs semantic search techniques, and utilizes LLMs for natural language interaction with users. The complementarity of these approaches offers significant advantages for developing LLM-based DSS.

3.1. CBR

One of the key application areas for KB DSS is supporting human decision-making during the operation of complex technical and socio-technical systems [35–37]. This includes, in particular, decision-making when a malfunction occurs at a technical facility, requiring a course of action best suited to the prevailing situation. An effective approach for building a DSS capable of proposing solutions based on in-depth situation analysis is CBR [38,39].

In CBR, the knowledge base of a KB DSS is organized as a library of cases, each describing an encountered situation and the corresponding solution applied. Thus, the cases contain descriptions of the actions necessary to resolve a problematic situation. The CBR cycle is illustrated in Figure 1.

The overall process of situational solution inference in CBR includes the following stages:

1. Identifying the current problematic situation occurring in the system;
2. Retrieving from the knowledge base a case that contains a situation most similar to the current one;
3. Extracting the solution from the retrieved case for reuse in the current situation;
4. Evaluating and, if necessary, adapting the extracted solution to fit the new context;
5. Transmitting the selected solution for execution;
6. Evaluating the outcome and retaining the new situation-solution pair in the knowledge base.

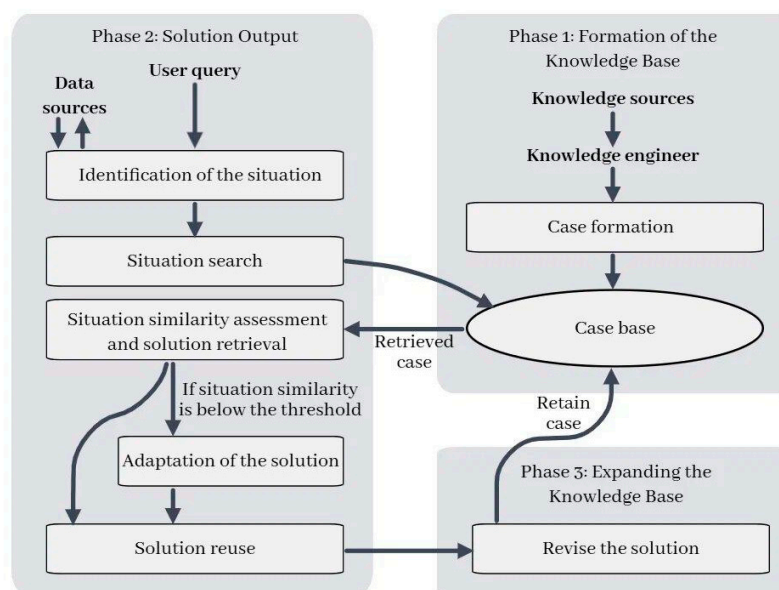


Figure 1. CBR cycle.

This approach ensures rapid solution inference from a repository of pre-existing solutions, which undergo rigorous verification during the creation and initial deployment phases. The primary challenge lies in formalizing case representation and implementing effective procedures for comparing and retrieving relevant problematic situations. In classical CBR, situations are represented using a set of attributes and their corresponding values, and various similarity metrics within the attribute space are employed for comparison and selection.

Concurrently, modern LLM/RAG technologies can be leveraged to describe situations and solutions in natural language, enabling solution inference through interactive dialogue and situational analysis with a human user.

3.2. RAG

LLMs utilizing RAG from a dedicated knowledge base enable human-DSS interaction through dialogue and allow unstructured texts to serve as the knowledge source. In response to a user query, relevant text segments are retrieved from the knowledge base using vectorization and text comparison algorithms. These segments may contain descriptions of the required actions, which is essential for KB DSS. However, unlike CBR, traditional RAG lacks a well-defined knowledge formalization model that explicitly links problem situations to their solutions at the level of quantitative parameters. Knowledge fragments in RAG are represented as unstructured or semi-structured texts, and the retrieval of relevant segments heavily depends on the configuration of the embedding models. These models often fail to account for the specific characteristics of the tasks or the constraints of a narrow subject domain.

A typical workflow of a system employing RAG is illustrated in Figure 2. It comprises three main steps: knowledge base preparation, information retrieval, and response generation. The naive RAG presupposes that source documents are segmented into chunks, embedded into vector representations, and subsequently stored in a vector database. During operation, the user query is vectorized, the most relevant fragments are retrieved from the knowledge base, and these fragments are incorporated into the prompt context. Based on this context, the LLM generates a response [40]. The adequacy and applicability of such a response in real-world decision-making processes depend on several factors, including the quality of the knowledge base, the accuracy of fragment

retrieval, and the effective utilization of these fragments by the model during response generation. In DSS, it is crucial to control these factors while preserving the flexible capabilities of the LLM. This enables effective AI-human interaction in decision-making processes, encompassing problem analysis, the selection of alternative actions, and the explanation and justification of the chosen solution.

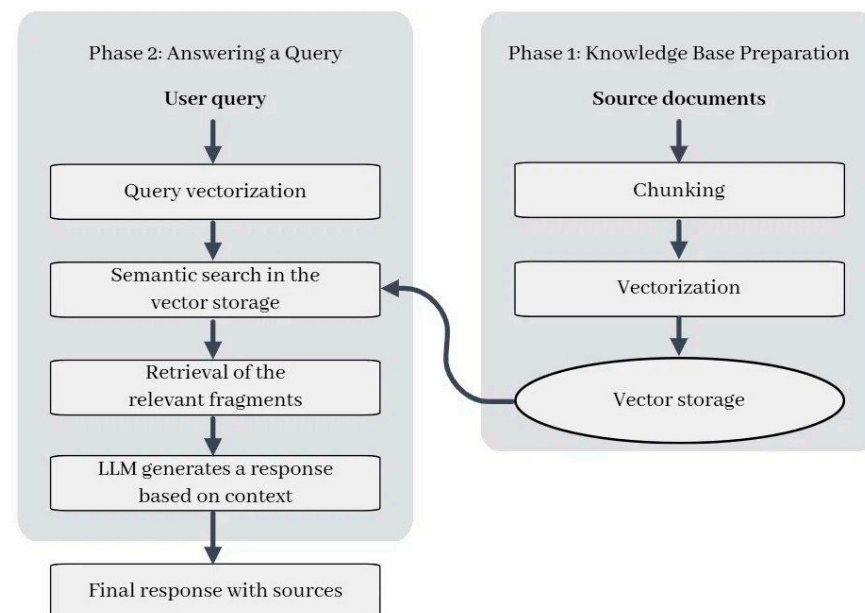


Figure 2. LLM + RAG Operation Scheme.

4. Concept

4.1. DSS Knowledge Base with RAG and CBR

Integrating RAG and CBR concepts offers dual advantages. On the one hand, it relieves developers from the labor-intensive procedures of attribute-based description of situations and solutions by allowing the use of natural language texts for this purpose. On the other hand, representing information in the knowledge base as well-structured cases enhances the accuracy of retrieving relevant knowledge, thereby making the DSS more reliable and efficient during situational decision-making.

This approach also establishes a unified template for knowledge description, which is crucial during the specialist interviewing and knowledge base construction stages to ensure the completeness, consistency, and non-contradictory nature of knowledge when resolving problematic situations. When developing a knowledge base for a DSS with RAG/CBR, the database is organized into cases rather than splitting texts into chunks (see Figure 2).

Another distinction lies in how the knowledge base is formed and its sources. RAG typically utilizes textual or text-graphical documents, whereas a KB DSS for decision-making relies on expert knowledge from specialists with significant practical experience in the subject domain. This knowledge is often undocumented and requires additional effort for its elicitation and representation [41].

When creating collaborative knowledge bases, the cases within the KB DSS become the result of collective expert work. To elicit this knowledge and represent it in a structured form within the knowledge base, we employ a template-based case representation defined as a set comprising

$$\text{Case} = \{\text{Event}, \langle \text{Cause}, \text{Symptoms}, \text{Action} \rangle_i \mid i = 1, 2, \dots\}, \quad (1)$$

where “Event” is a textual description of the external manifestation of a situation or an undesirable occurrence. The <Cause, Symptoms, Action> triplets serve to specify the situation by outlining possible causes with symptoms of the given event and the actions that need to be taken. <Actions> contain a description, in the form of semi-structured text, of the steps required to eliminate the cause and/or verify its presence.

The work of creating a collaborative KB is organized around a predefined list of events, i.e., cases. Such a list of events is typically known from practice and documented. However, it can also result from interactions among experts, including with the participation of an LLM.

4.2. LLM-Based Knowledge Engineering Approach

The task of a knowledge engineer in forming a knowledge base consists of eliciting knowledge through interaction with an expert and then encapsulating it in a formalized knowledge representation model.

It should be noted that an expert in decision-making tasks is a bearer of unique, non-formalized experience and, as a rule, a narrow specialist in a given subject domain. Therefore, in knowledge engineering and in expert assessment tasks for decision support, the term “experts” traditionally refers not to a broad pool of respondents, but to a small number of specially selected such specialists. In this paradigm, the quality of the knowledge elicited is ensured not by sample size, but by the depth of expertise and the quality of knowledge elicitation, which largely depends on the qualifications of the knowledge engineer.

In practice, the knowledge engineer’s task is considerably more complex than simply “eliciting” and “encapsulating” facts. The complexity lies in dealing with contradictions in expert knowledge: often, two highly qualified specialists may propose mutually exclusive heuristics for the same subject domain, each supporting their position with successful cases. In this situation, the engineer acts not merely as a passive “receiver” of information but as an arbiter and analyst. It is necessary not only to resolve the conflict but also to identify the conditions under which each strategy becomes valid, formalizing context-dependent rules; otherwise, the knowledge base will be internally inconsistent and inoperable.

Furthermore, working with the complementarity and semantic integration of cases becomes critically important. The engineer must address the challenge of semantic integration: distilling a unified representation from disparate, sometimes redundant, descriptions. The difficulty lies in the fact that the same entities may be referred to differently across cases, and explicit connections between events remain implicit in the expert’s reasoning. In practice, the knowledge engineer is compelled to engage in the re-engineering of thought processes, filling in missing logical links and unifying terminology to transform a collection of specific examples into a coherent, scalable model capable of producing correct conclusions beyond the original training cases.

In LLM-based knowledge engineering, it is assumed that the functions of a knowledge engineer can be largely transferred to a large language model [19,21]. The proposed LLM-based knowledge engineering approach, shown schematically in Figure 3, clarifies the place and role of LLMs in creating collaborative knowledge bases.

In this approach, each cycle of creating a collaborative knowledge base consists of the following stages:

1. Elicitation and formalization of individual knowledge, during which individual cases are formed;
2. Knowledge consolidation, during which a combined case is formed;
3. Knowledge Case evaluation for decision-making.

In this process, the first important concept is that of an individual case as a representation of the knowledge of a single expert. The second important concept is that of a consolidated case as a model for representing the collaborative knowledge of a group of experts.

In the first stage of knowledge elicitation, an LLM is used to interview experts using specialized prompts. A key feature of this approach is that the LLM acts as a structured dialogue facilitator, actively guiding the knowledge elicitation process through iterative questioning. In the second stage, the LLM aggregates the expert responses obtained earlier, using an aggregation-focused prompt.

At the evaluation stage, the functionality of the case base is verified. To this end, an evaluation methodology is developed that takes into account both the process and the outcome of decision inference (see Section 5.4). The resulting output can then be used for solution inference when the event described in the case occurs. Further expansion of the knowledge base with new cases can be organized without interrupting DSS operation.

An overall schematic of the proposed approach is shown in Figure 3.

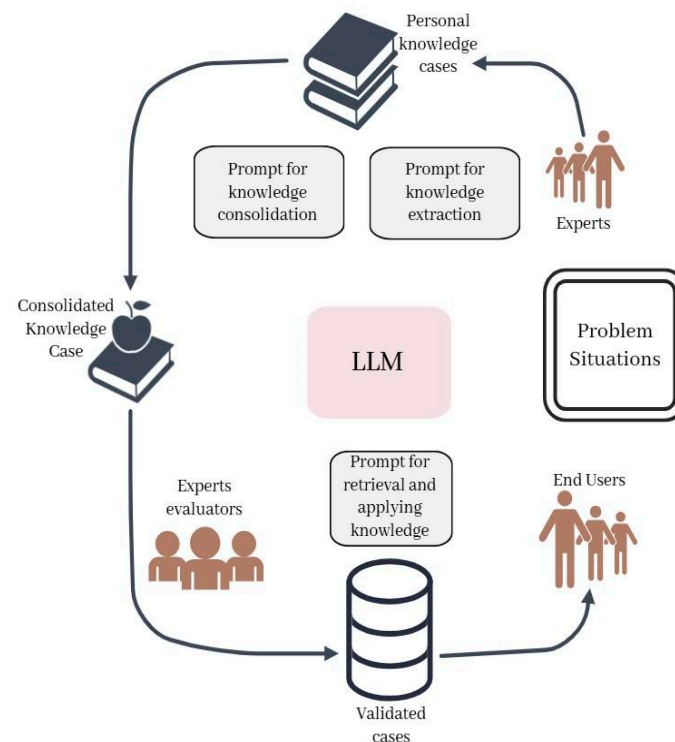


Figure 3. LLM-based knowledge engineering process for DSS collaborative knowledge bases.

4.3. Knowledge Elicitation and Consolidation with LLM-as-Knowledge-Engineer

According to the proposed approach, the implementation of the consolidated knowledge base formation procedure comprises two stages: elicitation of individual knowledge (personal knowledge cases) and their subsequent consolidation.

Tables 1 and 2 present the prompts for the knowledge elicitation task and the knowledge consolidation task, i.e., the merging of individual case descriptions obtained from experts.

In the template for the knowledge elicitation task (Table 1), the variable part is the description of the event for which the case is being formed, including causes, symptoms, and actions. Applying the prompt with the specified event enables the model, in dialogue with the experts, to form cases containing descriptions of possible causes and actions to be performed when the given event occurs. In this prompt, the user is designat-

ed as the expert with whom the LLM interacts through its interface during the process of eliciting and forming a knowledge case.

Table 1. Prompt with instructions for knowledge elicitation (Prompt_{extr}).

You create a knowledge base by having a dialogue with a person—a user. Knowledge in the knowledge base is presented as a chain: Event—Causes—Symptoms—Action, where Event means some event that happened, Cause is one of the possible causes of such an event, Symptoms is a description of how the cause manifests, and Action is the actions that need to be performed to react to this event. You create a knowledge base by asking the user questions. First, you ask about the causes that could have led to this event and make a list of possible causes. You ask for one cause using a question like this. Example of a question: An Event occurred; name the most likely possible cause of this event. After the user answers this question, you ask him the following: Are there any other possible causes? If he answers in the affirmative, you again ask the question about the possible cause that was specified above in this text. So you ask questions about causes until the user answers your question “Are there any other possible causes?” in the negative or says “I don’t know.” After that, you make a list of reasons for this event from all the named reasons. You include in this list only those reasons that the user mentioned. You do not add any other information about possible reasons to the list. Next, you take the first reason from the list of reasons and ask the user the following question: “Describe how the cause manifests.” You include the answer you receive from him in the list of Symptoms, and you associate this action with this reason. Next, you ask the user the following question: “Name the actions that need to be performed when an event occurs for this reason.” You include the answer you receive from him in the list of Actions, and you associate this action with this reason. You do this until the list of reasons is complete. After that, you format the knowledge base as a structured text, where the event, reasons and actions will be indicated, and display the entire knowledge base on the screen. Please note: during the dialogue, do not display intermediate results in the form of an already created structure for the knowledge base. Now I will ask an Event. Event = {Event content}. You start asking your questions to the user after I give you the next command “Start the dialogue.” It’s clear?

This prompt implements a structured dialogue algorithm in which the knowledge engineer (a role assumed by the LLM) strictly follows a linear extraction strategy: first, an exhaustive collection of hypotheses regarding the causes of the event is performed; then, for each cause, the corresponding actions are sequentially requested. The mechanism requires the expert to provide answers within the predefined framework of Event, Cause, Symptom, Action. In this process, the LLM acts not as an interpreter but as a coordinator, refraining from adding its own knowledge and strictly recording only the entities explicitly stated by the expert.

To elicit knowledge and formulate a case in dialogue with an LLM, we employ a formula for constructing the k -th case based on an expert interview S_k

$$\text{Case}_i = \text{LLM}(S_k, \text{Prompt}_{\text{extr}}), \quad (2)$$

where Prompt_{extr} is a knowledge elicitation prompt.

In the template for the knowledge consolidation task (Table 2), the variable component consists of the individual cases that were obtained in the previous stage.

Table 2. Prompt with instructions for case consolidation (Prompt_{cons}).

We have two knowledge bases called Knowledge Base 1 and Knowledge Base 2. Each knowledge base contains knowledge that is presented in the form of the Event—Cause—Symptoms—Actions structure. Causes are possible causes of an event, and actions are the recommendations and actions that need to be performed for this event and the cause associated with it. You need to combine two knowledge bases into one consolidated knowledge base. At the same time, you need to preserve the structure of the knowledge base. First, you need to analyze the causes from the two knowledge bases and make a new list of causes that will combine the lists of causes from these two knowledge bases. Keep in mind that the causes should not be repeated. If the causes are named differently, you need to replace them with one general name. Then you need to make a combined list of actions for each of the causes. To do this, you need to compare the actions that are listed in these knowledge bases opposite one cause and formulate a new list of actions. If for one cause the actions from different knowledge bases indicate the same thing, then you need to enter a new generalized content for this action. If these are actions that contradict each other, then in the consolidated knowledge base you will need to enter a generalized formulation of the action and say that a contradiction was identified. Take into account the clarifications and nuances in the steps, if any. Please note: when compiling a consolidated knowledge base, you do not include information from other sources; you use only these knowledge bases. These knowledge bases are shown below: Knowledge Base 1; {Knowledge Base 1}; End of Knowledge Base 1; Knowledge Base 2; {Knowledge Base 2}; End of Knowledge Base 2.
--

The consolidation operation, in terms of set theory, can be denoted as the union of individual cases received from different experts

$$\text{Case} = \text{Case1} \cup \text{Case2} \cup \dots \cup \text{CaseN}, \quad (3)$$

where N is the number of experts participating in the collaborative formation of the Case.

However, unlike a conventional union operation, it is necessary to account for the fact that the same concepts (such as events, causes, and actions) may be expressed through different texts across various cases. Different individuals describing the same cause or action will most likely use different wording.

Another important aspect of merging individual cases lies in potential contradictions. Solutions to the same problem may differ in the sequence of steps or be based on different algorithms.

This can pose a fundamental difficulty for forming a collaborative knowledge base using conventional methods. At the same time, LLM/RAG leverages mechanisms of vector text representations (embeddings), which are constructed taking into account semantic similarity. This allows the large language model to account for the semantic proximity of texts and introduce generalized formulations for concepts that were represented by different texts across cases, while simultaneously highlighting fundamental differences that need to be preserved for possible subsequent verification.

To this end, the prompt for the consolidation task includes an instruction for generalization in cases where the same cause or action appears in different formulations, along with a directive to identify and highlight contradictions, nuances, and refinements.

Thus, when using LLMs and appropriate instructional prompts, formula (3) is transformed into the following:

$$\text{Case} = \text{LLM}(\text{Case1}, \text{Case2}, \dots, \text{CaseN}, \text{Prompt}_{\text{cons}}) \quad (4)$$

where $\text{Prompt}_{\text{cons}}$ is a prompt with instructions for case consolidation.

To reduce context length, it is advisable to employ an iterative consolidation procedure, wherein only two cases are merged at each iteration. If N exceeds two, the result obtained at each iteration is then consolidated with the next individual case. This approach enables experiments to be conducted under constraints on the number of tokens per model query.

5. Results

5.1. Data Collection Approach

To conduct the experiments, five experts in residential engineering systems were interviewed. Knowledge was elicited regarding malfunctions in private homes, specifically focusing on: gas boiler error code F05, autonomous water treatment system failures, and autonomous water supply system malfunctions. Although the collected knowledge is not exhaustive, it is sufficient to preliminarily evaluate the effectiveness of the proposed knowledge base formation methodology based on the LLM-as-knowledge-engineer approach.

Each expert participated in two interviews. During the first interview, a knowledge base was constructed, with knowledge represented in a case format (situation and solution) as free-form text (Free-form Knowledge Base). An example of this question-and-answer format is provided in Table 3.

Table 3. An example of a question and answer in free form.

Question
Error F05 appeared on the Viessmann gas boiler and it shut down. What to do?
Answer
Error 05 on the boiler can indicate several different issues. First, it may be due to poor removal of exhaust gases. Check the flue pipe through which the exhaust gases are vented. It should not be iced over; any ice must be cleared away, including from inside the pipe. This ice forms from frozen condensate. Once you have cleaned the pipe, try restarting the boiler by unplugging it from the power outlet for 10 min; the error should reset itself. Second, this error can be caused by a faulty temperature sensor. In that case, the sensor will need to be replaced. To determine if the sensor is likely the problem, the proper method is to measure its resistance. Based on symptoms from my experience, in such cases the error typically appears immediately after a reboot or within 30 to 60 min of operation.

The interaction presented in Table 3 exemplifies an individual knowledge case.

During the second interview, the structured knowledge base was constructed using the LLM-as-knowledge-engineer methodology, in which the LLM conducted a survey of the expert by means of a specially designed prompt and presented the results in the form of a structured “Event–Cause–Symptoms–Action” model. The knowledge elicitation prompt is detailed in Table 1. An example of the resulting formalized individual case, following representation (1), is shown in Table 4.

Table 4. Example of a formalized individual case.

Event: F05 error
Cause: Fan (turbine) malfunction
Symptoms: Error 05 occurs once every few hours. If there is strong wind from the exhaust pipe side, the probability of shutdown increases. Exhaust gas does not exit the pipe.
Action:
1. Disconnect the gas and electricity supply to the boiler. Ensure the power is off.

2. Remove the boiler cover to access the fan.
3. If the boiler has failed to start more than 3 times in a row, before removing the fan itself, you must disconnect the D02 sensor.
4. Disconnect the fan power (the wire with the terminal is attached to the circuit board).
5. Manually check the fan: try to spin it. It should not rotate too stiffly, wobble, or hum.
6. If the fan rotates with difficulty, wobbles, or makes unusual noises, it must be cleaned or replaced. Cleaning, bearing lubrication, or complete fan replacement is required.
7. After checking/replacement, reconnect everything, turn on the power and gas, and start the boiler.

All participants were fully informed about the study's objectives and how their data would be used in scientific publications. Voluntary participation in the survey after receiving this information was considered sufficient confirmation of informed consent for this type of research.

5.2. Experimental Work. Consolidation Results

This section presents an example of consolidating a knowledge segment for a specific case in the knowledge base dedicated to gas boiler malfunctions in residential properties.

Gas boilers are equipped with error monitoring systems that display specific error codes in the event of a malfunction. This case examines a common error, F05, which causes a complete boiler shutdown and is among the most frequently occurring faults.

The F05 error represents a complex and representative scenario for decision-making regarding its remediation due to several factors.

First, this event can stem from various causes (e.g., condensate in the boiler pipes, flue blockage, or malfunctions of the fan, draft sensor, or electronic control board). Identifying these causes requires checking specific conditions and symptoms. For instance, equipment malfunctions may arise from the boiler's age, electronic boards can be damaged by power surges, and condensate may form under specific climatic conditions. Consequently, each potential cause manifests within a specific context, requiring targeted actions both to diagnose the cause and to eliminate the malfunction. An incorrect solution either fails to resolve the issue or leads to unnecessary diagnostic and repair costs. Moreover, existing information sources often lack a clear, precise action algorithm; successful troubleshooting relies heavily on the professional knowledge and empirical experience of the specialist. These features of the present case served as the basis for its selection in this pilot study. Although it represents a particular instance within a narrow problem domain, it exhibits typical characteristics of problematic situations that arise in production systems and are associated with complex decision-making scenarios.

The illustrative example in our study demonstrates the consolidation of three individual knowledge bases, merged sequentially. In Prompt_{cons} (Table 2), the inputs are labeled "Knowledge Base 1" and "Knowledge Base 2," where "Knowledge Base 1" is the result of prior consolidation and "Knowledge Base 2" is the newly introduced case.

Each fragment corresponds to interview data from a single expert. This illustrates a complex scenario where the action sequences proposed by three specialists for the same problem diverge at several steps. These divergences are non-contradictory and represent valid alternative solutions.

Table 5 presents the consolidation result for the F05 error caused by a turbine malfunction. The experimental results were obtained using the DeepSeek V3.1 model. We also conducted human consolidation experiments for comparison, involving both a domain specialist and an individual with no background in the subject area.

Table 5. The results of the consolidation of complementary knowledge bases for error F05 due to turbine malfunction.**Knowledge Base 1***Event:* F05 error*Cause:* Fan (turbine) malfunction*Symptoms:* Error 05 occurs once every few hours. If there is strong wind from the exhaust pipe side, the probability of shutdown increases. Exhaust gas does not exit the pipe.*Action:*

1. Disconnect the gas and electricity supply to the boiler. Ensure the power is off.
2. Remove the boiler cover to access the fan.
3. If the boiler has failed to start more than 3 times in a row, before removing the fan itself, you must disconnect the D02 sensor.
4. Disconnect the fan power (the wire with the terminal is attached to the circuit board).
5. Manually check the fan: try to spin it. It should not rotate too stiffly, wobble, or hum.
6. If the fan rotates with difficulty, wobbles, or makes unusual noises, it must be cleaned or replaced. Cleaning, bearing lubrication, or complete fan replacement is required.
7. After checking/replacement, reconnect everything, turn on the power and gas, and start the boiler.

Knowledge Base 2*Event:* F05 error*Cause:* Turbine malfunction*Symptoms:* Error 05 appears at intervals of once every few hours. During strong wind blowing from the exhaust pipe side, the probability of an emergency shutdown increases—exhaust gas stops exiting the pipe.*Action:*

1. Disconnect the gas and electricity supply to the boiler. Ensure the power is off.
2. Remove the boiler cover to access the fan.
3. Disconnect the tubes from the pressure sensor.
4. Perform a manual inspection of the fan: check the smoothness of the impeller rotation. It should rotate freely, without sticking, excessive shaft play, or unusual noise.
5. If a malfunction is detected: if rotation is stiff, there is significant play, or humming/grinding is heard, the fan requires cleaning, bearing lubrication, or complete replacement.
6. Completion of work: after troubleshooting, perform reassembly in reverse order, restore the electrical power and gas supply, then conduct a trial start of the boiler to check its operation.

Knowledge Base 3*Event:* F05 error*Cause:* Fan (turbine) malfunction*Symptoms:* Error 05 occurs intermittently. May be dependent on wind conditions. A fan rotation with unusual, grinding-like sounds can be heard.*Action:*

1. Disconnect the gas and electricity supply to the boiler.
2. Remove the boiler cover to access the fan.
3. Disconnect the tubes from the pressure sensor; do not touch the sensor itself.
4. Inspect the fan: the impeller should rotate without play or grinding.
5. If play is detected, the fan must be replaced, as it is not designed for disassembly and repair.
6. Install the new fan.
7. Clean the pipe housing on the fan housing side and center it using the three screws. It must not touch the fan itself.

Consolidated Knowledge Base By LLM*Event:* F05 error*Cause:* Fan (turbine) malfunction*Symptoms:*

Error 05 occurs at varying intervals (once every few hours, intermittently).

The probability of an emergency shutdown increases during strong headwinds from the exhaust pipe side.

Exhaust gas does not exit the pipe.

The fan may rotate with unusual sounds (humming, grinding).

Probable Cause: Malfunction of the boiler's fan (turbine).

Action:

1. Disconnect the gas and electricity supply to the boiler. Ensure the power is off.
 2. Remove the boiler cover to access the fan.
 3. Preparation for removal:
If the boiler has failed to start more than 3 times in a row — disconnect the D02 sensor.
Disconnect the pressure sensor tubes without detaching the sensor itself (unless its replacement is required).
 4. Disconnect the fan power supply (the wire with the terminal on the circuit board).
 5. Manually check the smoothness of the impeller rotation. Rotation should be free, without sticking, excessive shaft play, humming, grinding, or other unusual sounds.
 6. Actions if a malfunction is detected:
If problems are identified (stiff rotation, play, noise):
ATTENTION: Different recommendations are provided in the source materials.
Option 1 (Sources 1, 2): Perform cleaning and bearing lubrication if possible.
Option 2 (Source 3): Replace the fan if it is non-serviceable or has play.
It is recommended to clarify the fan model and the manufacturer's instructions.
 7. If a new fan is installed or after cleaning: clean the pipe housing on the fan housing side and center it using the three screws. The housing must not touch the fan.
 8. Perform reassembly in reverse order (reconnect the pressure sensor tubes, fan wire, D02 sensor if necessary).
 9. Restore the electrical power and gas supply.
 10. Perform a trial start of the boiler to check its operation.
-

Consolidated Knowledge Base By Specialist

Event: F05 error

Cause: Fan (turbine) malfunction

Symptoms: Error 05 occurs periodically. It may depend on wind strength. If there is strong wind blowing from the pipe side, the probability of occurrence is higher. Grinding or crunching sounds may be heard in the fan area. Gases are not being properly vented.

Actions:

1. Disconnect the gas and electricity supply to the boiler. Ensure the power is off.
 2. Remove the boiler cover to access the fan.
 3. Disconnect the tubes from the pressure sensor (sensor D02); do not touch the sensor itself.
 4. If the boiler has failed to start more than 3 times in a row, the D02 sensor must be disconnected from the board.
 5. Disconnect the fan power (the wire with the terminal is attached to the circuit board). Remove the fan.
 6. Perform a manual inspection of the fan: check the smoothness of the impeller rotation. It should rotate freely, without sticking, excessive shaft play, or unusual noise.
 7. If a malfunction is detected: if rotation is stiff, there is significant play, or humming/grinding is heard — cleaning of the fan and bearing lubrication are required. However, not all fan models are serviceable; in such cases, it is simpler and more reliable to replace the fan.
 8. Clean the pipe housing on the fan housing side and center it using the three screws. It must not touch the fan itself.
 9. Completion of work: after troubleshooting, perform reassembly in reverse order, restore the electrical power and gas supply, then conduct a trial start of the boiler to check its operation.
-

Consolidated Knowledge Base By Non-Specialist

Event: F05 error

Cause: Fan (turbine) malfunction

Symptoms: Error 05 occurs once every few hours. When there is strong wind from the exhaust pipe side, the probability of an emergency shutdown increases; exhaust gas stops exiting the pipe. The malfunction is accompanied by sounds resembling fan rotation grinding.

Actions:

1. Disconnect the gas and electricity supply to the boiler. Ensure the power is off.
2. Remove the boiler cover to access the fan.
3. If the boiler has failed to start more than 3 times in a row: before removing the fan itself, disconnect the tubes from the D02 pressure sensor; do not touch the sensor itself.
4. Disconnect the fan power (the wire with the terminal is attached to the circuit board).
5. Perform a manual inspection of the fan: check the smoothness of the impeller rotation. It should rotate freely, without sticking, excessive shaft play, or unusual noise (humming, grinding).
6. If a malfunction is detected—stiff rotation, humming/grinding, play—the fan must be completely replaced, as it is not designed for disassembly and repair.
7. Install the new fan.
8. Clean the pipe housing on the fan housing side and center it using the three screws. It must not touch the fan itself.
9. Restore the electrical power and gas supply, then conduct a trial start of the boiler to check its operation.

As shown in Table 5, personal knowledge bases 1 and 2 contain differences in steps 3 and 4, while base 3 includes a unique step for centering the pipe casing. These steps do not contradict each other and can be considered mutually complementary. No semantic discrepancies are observed in the remaining steps.

The consolidated knowledge base produced by both the LLM and the domain specialist incorporates the distinct and common steps from the individual bases. In contrast, the base produced by the non-specialist contains inaccuracies at steps 3 and 6 (regarding the disconnection of the sensor and the repair of fan bearings).

This example demonstrates that the LLM can effectively combine cases, incorporate missing concepts, and generalize formulations when necessary. At the consolidation phase, the LLM does not assume the role of verifying or conclusively validating individual cases; rather, it functions as an instrument for detecting inherent contradictions and substantive inconsistencies embedded within them (see the warning note in the example of the consolidated knowledge base by the LLM in Table 5). Furthermore, the LLM requires substantially less time to consolidate two cases (10 s) compared to the human specialist (5–7 min) and the non-specialist (10–20 min).

However, drawing definitive conclusions requires a detailed, quantitative evaluation of the created case base's functionality. To this end, we tested the knowledge base in a decision inference mode using the prompt described in Section 5.3. Within the framework of a human evaluation approach [42], we developed a methodology for expert assessment of decision inference results. Using this methodology, we obtained quantitative estimates and compared the inference results with and without the use of the created knowledge base. The evaluation methodology and the results of this comparative study are presented in Sections 5.4 and 5.5, respectively.

5.3. Decision Retrieval Prompt

The general formula for inferring a solution (Sol) in a given situation using the created knowledge base KB is as follows:

$$\text{Sol} = \text{LLM}(\text{KB}, \text{Prompt}_{\text{des}}). \quad (5)$$

where $\text{KB} \supseteq \text{Case}$; $\text{Prompt}_{\text{des}}$ is the prompt used to conduct a dialogue with the user in order to obtain information about the problem situation according to the case structure (1) and to infer a recommended solution.

The $\text{Prompt}_{\text{des}}$ for the problem-solving inference task is presented in Appendix A. At this stage, the user's role specified in $\text{Prompt}_{\text{des}}$ is no longer the expert (as in the knowledge base formation stages), but rather the end-user of the KB DSS who is seeking to resolve the problem.

The operational methodology comprises an initialization phase, during which the LLM agent awaits a strictly defined user command (“Start”) for activation. This is followed by symptom gathering through guided questions to assist the user in detailing the problem description. Subsequently, matching and inference are performed: symptoms are matched against entries in the KB, and upon a successful match, the name of the situation (fault) and the corresponding solution algorithm are provided verbatim from the KB. The process is iterative: if the initial solution does not resolve the issue, the agent sequentially proposes alternative causes and solutions guided by the KB. The operation concludes based on strict criteria: successful problem resolution, exhaustion of options within the KB, or a direct user command.

5.4. Methodology for Evaluating Knowledge Base Performance in Solution Inference

The results of solution inference in an RAG system are influenced not only by the content of the knowledge cases but also by the model’s ability, guided by control prompts, to retrieve the relevant case, clarify missing information through user dialogue, and infer the prescribed solution.

To evaluate the overall performance of the developed knowledge base using a human evaluation approach, a methodology was developed in which external expert evaluators (EE) assess two aspects: the dialogue process during solution inference and the final inference outcome. These evaluators were domain specialists not involved in the knowledge base development.

This methodology enables the assessment of system performance in the absence of unambiguous ground-truth answers, which is particularly important for situations characterized by high uncertainty and variability in achieving the desired result.

The metric set reflects classical requirements for the completeness and accuracy of inferred solutions, while also incorporating findings and recommendations for multi-dimensional human evaluation dimensions presented in [43,44].

Tables 6 and 7 present the metrics and instructions provided to the experts for scoring.

Table 6. Assessment of the Dialogs Score.

Criterion	Description	Assessment
Appropriateness of the Question	The model’s interrogatives throughout the dialogue are comprehensible and contextually relevant to the query (the causal–logical relationship between questions and responses is explicitly established)	0–10
Quality of Messages (Inter-mediate Responses)	The model’s utterances are clear, unambiguous, and contextually appropriate	0–10
Aggregate Assessment of the Result Acquisition Method (Dialogs Score)	Computed by summing the individual criterion scores and normalising through division by the maximum attainable value (20)	0–1

Note: The following reference points are defined for the 0 to 10-point scale: 10 points (extremely high), 7 points (fairly high), 5 points (moderate/acceptable), 3 points (low, requires improvement), and 0 points (extremely low).

Table 7. Assessment of the Solution Score.

Criterion	Description	Assesment
Adequacy (Problem Relevance)	The underlying cause has been correctly identified (the problem has been accurately and comprehensively diagnosed)	0–10
Completeness	The proposed solution demonstrates completeness—it encompasses all actions requisite for the resolution of the problem	0–10
Precision	The proposed solution exhibits precision—no incorrect or superfluous (i.e., contextually irrelevant) actions are included	0–10
Applicability	The proposed solution features a clearly articulated, comprehensible, and practically implementable action plan suitable for deployment in the given situational context	0–10
Solution Score	Computed by aggregating the individual criterion scores and normalising through division by the maximum attainable value (40)	0–1

Note: The following reference points are defined for the 0 to 10-point scale: 10 points (extremely high), 7 points (fairly high), 5 points (moderate/acceptable), 3 points (low, requires improvement), and 0 points (extremely low).

A comprehensive assessment of the system's performance, assuming equal equivalence of the two considered aspects (result quality and method of obtaining the result), is calculated as follows:

$$QS = \frac{(\text{Solution Score} + \text{Dialogs Score})}{2} \quad (6)$$

Or in the more general case

$$QS = \alpha * \text{Solution Score} + \beta * \text{Dialogs Score}, \quad (7)$$

where α and β are relative importance weights.

When evaluated by multiple experts, the average QS value is calculated.

In this assessment, we assume that a knowledge base consisting of verified cases necessarily contains the solution required by the user; thus, the primary challenge remains the retrieval, interpretation, and application of the relevant case by the language model.

5.5. Experimental Work. Comparison Solution Inference

This section provides a comparative analysis of the inferred solutions obtained using a knowledge base built according to the proposed methodology versus those derived from a free-form knowledge base.

The experiments followed the scheme shown in Figure 4. The user was provided with a set of cases, each comprising a problem and associated symptoms, from which a solution was to be derived. This set was input into the LLM + RAG system using three different knowledge bases in turn: a Free-Form Knowledge Base (a fragment of which is provided in Appendix B), a Structured Non-Consolidated Knowledge Base (fragments presented in Table 4), and a Structured Consolidated Knowledge Base (fragments presented in Table 5). Three different LLMs were investigated: DeepSeek (DeepSeek V3.1, 37 billion parameters, context length 128K, temperature 0.8), Mistral AI (Mistral Large 2, 123 billion parameters, context length 128K, temperature 0.7), and Qwen (Qwen 3.5, 122 billion parameters, context length 256K, temperature 0.7), accessed via their official web interfaces.

Consequently, nine experiments were conducted for each case (three knowledge bases with three LLMs).

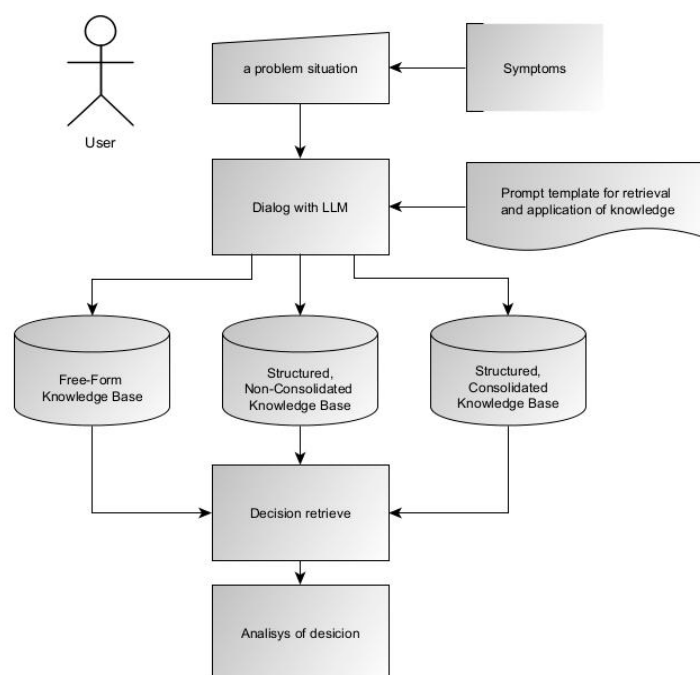


Figure 4. The scheme of the decision retrieve experiment.

Table 8 presents a comparison of solution inference from the three knowledge bases using the DeepSeek V3.1 model. The prompt described in Section 5.3 (Appendix A) was used for knowledge inference.

The comparison was performed using an identical set of symptoms: Error F05 appears intermittently, wind affects its frequency, and the vent pipe is not iced up.

Table 8. Comparison of the solution composition in response to the fan problem.

Solution Inference: Free-Form Knowledge Base

1. Reset the boiler. You may need to restart it several times for it to resume normal operation and clear the error.
2. Go outside and inspect the exhaust pipe. Ensure the pipe is tilted slightly away from the wall so condensate can freely drain outward.
3. Check for icicles on the pipe. Make sure the bell end is not completely frozen (this can happen during heavy winds and snowfall).
4. Remove any ice present inside or blocking the pipe.

Solution Inference: Structured, Non-Consolidated Knowledge Base

1. Turn off the gas and electricity supply to the boiler. Make sure the power is disconnected.
2. Remove the boiler cover to access the fan.
3. Disconnect the tubes from the pressure sensor. Do not touch the sensor itself.
4. Inspect the fan: Check that the impeller rotates without play or scraping.
5. If play (looseness) is detected, the fan must be replaced because it cannot be disassembled or repaired.
6. Install a new fan.
7. Clean the housing of the pipe from the fan housing side and center it using the three screws. It must not touch the fan itself.
8. After completing the work, reassemble in reverse order, restore gas and power, and perform a test run of the boiler to check its operation.

Solution Inference: Structured, Consolidated Knowledge Base

1. Shut off the gas and electricity supply to the boiler. Ensure that the power is completely disconnected.
2. Remove the boiler cover to gain access to the fan.
3. Preparation for Demounting:

-
- If the boiler has failed to start more than 3 times consecutively, disconnect sensor D02.
Disconnect the pressure sensor tubes without removing the sensor itself (unless its replacement is required).
4. Disconnect the power supply to the fan (the wire with the terminal on the circuit board).
 5. Manually check the smoothness of the impeller's rotation. Rotation must be free, without jamming, excessive shaft play, humming, grinding, or any other unusual noises.
 6. Actions in Case of Malfunction:
If problems are identified (difficult rotation, play, noise):
ATTENTION: The source materials provide differing recommendations.
Option 1: Perform cleaning and lubrication of the bearings, if possible.
Option 2: Replace the fan if it is non-serviceable or has play.
It is recommended to clarify the fan model and consult the manufacturer's instructions.
 7. If a new fan is installed or after cleaning: clean the pipe housing on the fan housing side and center it using three screws. The housing must not touch the fan.
 8. Perform reassembly in reverse order (reconnect the pressure sensor tubes, the fan wire, and sensor D02 if necessary).
 9. Restore the supply of electricity and gas.
 10. Conduct a test startup of the boiler to verify its operation.
-

Analysis of the responses reveals that for the unstructured knowledge base, the solution composition is generally consistent but omits several critical details. It fails to specify that the D02 sensor should only be touched after unsuccessful ignition attempts, omits the potential need for fan disassembly, and lacks the step regarding cleaning and centering the pipe casing.

For the structured, non-consolidated knowledge base, only the first expert's solution composition is inferred. There is no mention of the possible need to disconnect the D02 sensor, and distinctive steps from other experts are only generated upon additional queries.

At the same time, as can be seen from Table 8, the solution from the structured consolidated database contains recommendations from all experts. This confirms the expediency of combining personal cases with a separate preparatory stage.

Thus, structuring and consolidating expert knowledge prevents information loss and ensures the solution is inferred with all necessary refinements.

Table 9 presents the model comparison results according to the methodology outlined in Section 5.4 (Formula 6). The evaluation was conducted by three expert evaluators, and the table shows the final averaged results.

Table 9. Results of model comparison when making decisions.

Free-Form Knowledge Base			
	Solution score	Dialogs Score	QS
DeepSeek	0.35	0.47	0.41
Mistral AI	0.25	0.43	0.34
Qwen	0.3	0.43	0.365
Structured, Non-Consolidated Knowledge Base			
	Solution score	Dialogs Score	QS
DeepSeek	0.65	0.60	0.625
Mistral AI	0.6	0.60	0.6
Qwen	0.65	0.57	0.61
Structured, Consolidated Knowledge Base By LLM			
	Solution score	Dialogs Score	QS
DeepSeek	0.73	0.75	0.74

Mistral AI	0.7	0.735	0.72
Qwen	0.73	0.73	0.73
Structured, Consolidated Knowledge Base by Specialist			
	Solution score	Dialogs Score	QS
DeepSeek	0.80	0.80	0.8
Mistral AI	0.77	0.77	0.77
Qwen	0.80	0.77	0.79
Structured, Consolidated Knowledge Base by Non-Specialist			
	Solution score	Dialogs Score	QS
DeepSeek	0.65	0.75	0.7
Mistral AI	0.6	0.735	0.67
Qwen	0.65	0.735	0.69

The evaluation revealed similar deficiencies in the performance of the three models. For the Free-Form Knowledge Base, over-generalization of the formulations caused a loss of critical information. For the Structured Non-Consolidated Knowledge Base, the response retrieved only a single case relevant to the problem, thereby omitting the information from the remaining cases. For the Structured Consolidated Knowledge Base, all models successfully output the complete relevant case.

Another notable observation is that cases developed by the domain specialist yielded evaluation results comparable to those of the LLM (QS: 0.77–0.80 vs. 0.72–0.74). In contrast, cases prepared by the non-specialist exhibited a lower QS of 0.67–0.70.

6. Discussion

6.1. Analysis of the Results

The elicitation, formalization, and representation of expert knowledge for use in inference algorithms remain a key challenge in the field of knowledge-based systems. Knowledge engineering, a discipline established to address this issue, has proposed various methods and models for knowledge representation, including frames, inference rules, and network models [43,45]. However, the process of creating knowledge bases and knowledge-driven inference algorithms remains extremely costly and labor-intensive, often requiring substantial time and human effort.

The idea of automating knowledge engineering activities using LLMs is highly appealing, as it could reduce the costs of building knowledge bases and potentially drive substantial changes in knowledge engineering practices [21]. In this pilot study, we proposed an approach to implement this concept and tested its feasibility using a technical case study. This case involves a complex problem situation characterized by selection ambiguity and a wide range of potential solutions.

In our experiments, LLMs utilizing specially designed prompts successfully acquired knowledge from experts, consolidated it, and organized it into a structured format. We also conducted an experimental evaluation of the acquired knowledge in the context of solution inference. During this process, a dialogue was established between the LLM and a user, who answered the model's questions to identify the current situation and ultimately received the required solution. During testing, these users acted as external expert evaluators.

We analyzed the results by comparing two knowledge engineering approaches: the traditional approach involving a human knowledge engineer (human-based approach) and the proposed LLM-based approach. To structure this analysis, we focused on the following aspects.

Quality of the created knowledge base. Evaluation by expert evaluators demonstrated that an LLM paired with a knowledge base structured as consolidated cases achieves higher performance in decision inference compared to an unstructured, non-consolidated knowledge base. However, this does not imply that the LLM-based approach guarantees superior results compared to the human-based approach. The results in Table 9 indicate that a consolidated case generated by the model yields better decision inference metrics than a case consolidated by a non-specialist. Nevertheless, a case consolidated by a domain specialist achieves higher QS values.

In practice, the quality of the knowledge base depends heavily on the selected experts, the qualifications of the knowledge engineer (in the human-based approach), and the chosen model and prompts (in the LLM-based approach). In either case, testing and trial operation are necessary to validate the constructed knowledge base.

Time costs. We assume that the time required for selecting experts, physically implementing the knowledge base, testing it, and conducting trial operations will be similar for both approaches. Time savings are primarily achieved in the processes of eliciting and consolidating knowledge cases. In our experiments, the time required to form an individual case through dialogue with an LLM was approximately 3 to 5 min, depending largely on the interviewed expert. Notably, a human knowledge engineer did not directly participate in this procedure. The time required for case consolidation using an LLM is determined by technical factors—including model parameters, network latency, and software interface performance—and amounted to approximately 10 s. In contrast, human attempts at knowledge consolidation proved to be highly labor-intensive. This process required not only significant time (4 to 6 min for a specialist and 15 min for a non-specialist to consolidate two cases) but also domain expertise. This latter requirement poses a particularly significant obstacle to creating collaborative knowledge bases, as it would necessitate involving experts not only for knowledge elicitation but also for consolidation.

It should also be noted that implementing an LLM-based approach, like any automation of human activity, incurs costs for developing and testing prompts and application software. These costs are justified if the developed solutions are subsequently replicated across a wide range of recurring implementation scenarios.

Qualification requirements. The traditional human-based approach demands a high level of qualification from the knowledge engineer, necessitating the involvement of professionals who specialize in expert knowledge elicitation.

Conversely, the LLM-based approach requires specialists to design, test, and tune prompts for specific problem domains, thereby introducing new qualification requirements for knowledge base developers. It is unlikely that LLM-based systems engineering will completely eliminate the knowledge engineer from the development process. However, it will likely transform the nature of their work, shifting the focus from direct human communication to organizing AI-human interaction and prompt engineering.

6.2. Study Limitations

This research is a pilot study, limited by the small volume of data and the limited number of cases tested within a single problem domain. While the obtained results support the feasibility of the LLM-based approach, they do not confirm its general applicability.

The present study did not address the issues of selecting and forming expert groups for knowledge base construction, although these issues may be of critical importance in the development of a DSS. Likewise, the current study did not address the identification of contradictions or the reconciliation of expert opinions during knowledge base creation and functionality evaluation. These are traditional challenges in knowledge engineering

and expert evaluation. While applying LLMs to these issues shows promise, identifying contradictions and evaluating the consistency of expert opinions were beyond the scope of this stage and will be addressed in subsequent research on the LLM-based approach.

Another limitation is the use of simplified external evaluation procedures to verify the knowledge base being created. While both human evaluation and the LLM-as-a-judge approach are used to evaluate LLM outputs [46,47], our pilot study relied exclusively on human experts. For future full-scale research, we plan to develop a prompt system and implement the LLM-as-a-judge approach using AI agents. This would enable more extensive testing across a larger number of cases, yielding statistically significant results and allowing for more robust conclusions regarding the proposed approach.

A further concern raised by the reviewers was the need to investigate critical issues such as the safety of technical systems. Due to the inherent characteristics of LLMs, they may be prone to hallucinations or inaccuracies even within RAG-based systems, and the prompts employed have a significant impact on this [24]. Consequently, the troubleshooting solutions proposed by an LLM can affect both system functionality and environmental safety. Therefore, in LLM-based decision support systems (DSS), alongside the traditional verification and validation procedures, it becomes essential to also assess the suitability of prompts and select the appropriate language model.

Importantly, we emphasize that the proposed approach is not yet ready for direct operational deployment. Any practical application must be preceded by expert verification, strict compliance with manufacturer documentation and applicable safety regulations, and comprehensive field testing under real-world conditions. Until such validation has been completed, the approach should be regarded as a research prototype rather than a production-ready tool. A systematic analysis of the security issues associated with LLM-based decision support systems thus remains a key direction for future work.

In summary, the LLM-based approach has demonstrated its feasibility for constructing a knowledge base within a specific subject domain. It holds the potential to reduce the time required to create knowledge bases, especially collaborative ones, and can lower the qualification requirements for knowledge engineers regarding direct expert communication. However, the practical implementation of this approach requires further research and experimentation to develop a robust methodology and a cost-effective technology for LLM-based knowledge engineering.

6.3. Directions for Future Development

Large language models facilitate the relatively straightforward creation of question-answering systems [44]. However, the capabilities of LLMs extend far beyond simple question answering. By leveraging domain-specific knowledge bases and RAG technology, LLMs can propose solutions in complex situations at the level of a domain expert, while also justifying and explaining their conclusions and recommendations in professional, comprehensible language.

These exact capabilities were envisioned in the classical architecture of expert systems, which remain a quintessential example of knowledge-based systems [43,48]. The practical implementation of explainability requires the development and investigation of appropriate prompts and AI-human interaction algorithms. In this regard, the concept of LLM agents configured to perform specific tasks using dedicated prompts and tools is highly promising [49]. Specifically, distinct agents can be designed for knowledge elicitation, consolidation, solution verification, and explanation. This research direction aligns with the frontiers in machine learning and knowledge extraction identified by A. Holzinger, L. Longo et al. in [50], specifically Frontier 04: Foundation Models, LLMs, and AI Agents.

Utilizing structured knowledge representation enables more accurate and faster inference compared to using unstructured texts in vector database-driven question-answering systems [51]. In this study, we employed a case-based structure for knowledge representation, encompassing descriptions of events, causes, and actions. While this serves as a foundational model for knowledge structuring, it is not the only viable option. A comparison with other knowledge representation methods, as well as with emerging RAG frameworks (e.g., Lightrag [52], AtomRAG [53]), lies beyond the scope of this study. However, future developments in LLM-based knowledge engineering may incorporate other structuring models, including knowledge graphs, ontologies, and production rules. The choice of a specific knowledge representation model will likely be dictated by the characteristics of the problem domain, the specific tasks, and the operational conditions for which the DSS is being developed.

Concurrently, critical attention must be paid to investigating and mitigating issues such as hallucinations and hallucinatory generalizations in language models.

We also identify potential for applying the LLM-based approach to develop novel systems for assessing the knowledge and qualifications of students and professionals. This is particularly important for training operational and dispatch personnel in complex technical systems to make decisions during undesirable or emergency events. Such personnel include air traffic controllers, power plant operators, pipeline dispatchers, and similar roles. Currently, LLMs are already being used to assess student knowledge through testing, evaluating written work, and analyzing responses [54,55]. The LLM-based knowledge engineering approach enables learners to create their own individualized knowledge bases. Subsequently, evaluating these knowledge bases by comparing them with expert knowledge or by assessing their application to problem-solving could establish a novel method for evaluating professional qualifications. Thus, a highly promising direction emerges for future research in the field of LLM-based expert knowledge systems.

7. Conclusions

The main results of this study are summarized in the following conclusions.

We proposed an LLM-based knowledge engineering approach that streamlines the elicitation, formalization, and integration of expert knowledge to create collaborative knowledge bases.

To evaluate the proposed approach, we conducted a pilot study and developed a model for structuring textual knowledge representation using the “Event–Cause–Symptom–Action” framework. Furthermore, we designed and tested specialized prompts for interviewing domain experts, generating individual knowledge cases, and consolidating them into a unified knowledge base.

To assess the functionality of the constructed knowledge base, we proposed an evaluation methodology based on its application for decision inference during an interactive dialogue between an LLM and an expert evaluator. We also developed an end-to-end implementation of the proposed approach using a technical case study, which confirmed the feasibility of the “LLM-as-knowledge-engineer” concept for solving practical problems.

The novelty stems from the presented LLM-based approach for creating collaborative knowledge bases. It includes the model representation of knowledge as structured cases (Event–Cause–Symptoms–Action) coupled with a procedure for knowledge elicitation and consolidation. This integration enables retrieving more accurate decisions from the knowledge base while accounting for potential contradictions in experts' knowledge. Without integration and formalisation into the proposed structure, knowledge may be lost during the decision retrieval process. The potential efficiency of

the approach is demonstrated by the results in Table 8, the evaluation represented in Table 9, and the illustrative examples in Supplementary Materials.

However, it is worth emphasizing that the results indicate feasibility rather than confirm the broad applicability of the proposed approach. The results are not statistically conclusive. Further investigation should include assessment of standard deviations, confidence intervals, inter-rater agreement, and significance testing.

In general, this study once again demonstrates that the emergence of modern LLM systems may mark a new stage in the development of intelligent systems based on expert knowledge, capable of solving time-consuming and practically significant tasks through effective human-AI interaction.

Supplementary Materials: The following supporting information can be downloaded at: https://github.com/diglukhikh/collab_kb_experimental/blob/171c41b7fcc9a65c9699bdf714e5d1e71e84c93/links accessed on 28 May 2026) This repository contains experimental chat logs from the DeepSeek and Qwen AI models, documenting the consolidation of complementary knowledge bases for diagnosing error F05 (turbine malfunction). The experiments encompass: (1) the merging of individual cases #1, #2, and #3; (2) the resulting final consolidated knowledge bases; and (3) a comparative analysis of decision retrieval based on three configurations: the specialist-driven knowledge base, the free-form knowledge base, and the LLM-consolidated knowledge base. These materials provide the raw data necessary for evaluating knowledge integration and retrieval performance in technical troubleshooting domains.

Author Contributions: Conceptualization, I.G. and D.G.; methodology, I.G. and D.G.; formal analysis, I.G.; data curation, D.G. and K.G.; experiment conduction, D.G. and K.G.; writing—original draft preparation, I.G.; writing—review and editing, I.G. and D.G.; visualization, K.G. and I.G. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the Ministry of Science and Higher Education of the Russian Federation within the framework of a State assignment, grant number FEWZ-2024-0052.

Data Availability Statement: The original contributions presented in this study are included in the article/Supplementary Materials. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LLM	Large Language Model
DSS	Decision Support Systems
KB	Knowledge Base
KB DSS	Knowledge-Based Decision Support Systems
RAG	Retrieval-Augmented Generation
CBR	Case-Based Reasoning
NLP	Natural Language Processing
NER	Named Entity Recognition

Appendix A. Decision Retrieval Prompt

You are a gas boiler repair expert. Your task is to determine the cause based on the user's description and suggest steps to resolve it, strictly relying only on the knowledge base provided below.

You must identify the problem both by the description of symptoms or manifestations and by a similar description of an event. Critical operating rules:

1. It is forbidden to use any sources of information other than the specified knowledge base.
2. It is forbidden to invent technical details, parameters, part names, or diagnostic methods not present in the base.
3. It is forbidden to give recommendations not confirmed by points in the knowledge base.
4. If the base has no data on the user's question, give the answer: "Unfortunately, there is no data on this question in the knowledge base."
5. It is forbidden to write anything except the initial phrase before the user starts the dialogue!
6. When communicating with the user, do not mention the existence of the knowledge base.
7. If the user described symptoms or a situation (problem), write the name of that situation according to the knowledge base.

After receiving the prompt, write the initial phrase: "All clear, let's begin!" To start the dialogue, write: "Start."

Procedure only after the user's command:

Reply: "Describe your problem."

8. Ask the user for a problem description with leading questions to be able to provide one specific solution from the knowledge base.
9. After the problem description: a. Match the symptoms with the list of errors from the base. b. Select the corresponding cause and write out the solution steps. c. If the user says the problem is not resolved, then ask leading questions about another cause. Then propose its solution. d. Ask questions until the causes and solutions in the database are exhausted or the user says the problem is resolved.
10. If the user says the problem is resolved, then end the dialogue: "Can I help with anything else?"
11. If the causes and solutions in the knowledge base are exhausted and the problem is not resolved, say: "Unfortunately, I cannot solve your problem!"
12. Upon a request to end the dialogue, immediately reply: "All the best!"
13. In response to the user's thanks, immediately reply: "All the best!"

Knowledge base (the only permitted source): {Knowledge base}

Note: In the pilot study, the Prompt_{des} was utilized to process a limited set of events (specifically, the F05 error). In a production deployment of the KB, this prompt should be augmented with an event identification module and corresponding rules. This enhancement will enable the LLM agent to accurately locate the case corresponding to a specific event within the knowledge base and engage in an interactive dialogue to clarify symptoms and potential causes.

Appendix B. Experimental Non-Structured Knowledge Base

Base 1: Error 05 on the boiler. Signs that the turbine is faulty: if the error appears once every few hours, or if there is strong wind blowing directly into the exhaust pipe. What to do: turn off the gas and electricity, then remove the boiler cover. If the boiler has failed to start more than three times in a row, before removing the fan itself, disconnect the D02 sensor and detach its wire from the control board. Manually spin the impeller. It is important that it does not feel stiff, wobble on the shaft, or produce a humming sound. If any of these issues are present, the impeller needs to be either cleaned or replaced. After that, reassemble everything in the reverse order. Restore power and gas supply; the boiler should start, though possibly not on the first attempt.

Base 2: Error 05 on the boiler. Signs that the turbine is faulty: when the wind intensifies from the direction of the exhaust pipe, and the error occurs every few hours. How to remove the fan: turn off the gas and electricity, then remove the cover. Disconnect the tubes from the pressure sensor. Next, inspect the fan. Its rotation should be smooth, without any play or grinding noise. If the rotation is stiff, if there is play, or if you hear a hum or grinding, then the bearings need cleaning and lubrication, or the entire fan must be replaced. After completing these steps, reassemble all components, turn the electricity and gas back on, and start the boiler.

Base 3: Error 05 on the boiler can indicate several different issues. First, it may be due to poor removal of exhaust gases. Check the flue pipe through which the exhaust gases are vented. It should not be iced over; any ice must be cleared away, including from inside the pipe. This ice forms from frozen condensate. Once you have cleaned the pipe, try restarting the boiler by unplugging it from the power outlet for 10 min; the error should reset itself. Second, this error can be caused by a faulty temperature sensor. In that case, the sensor will need to be replaced. To determine if the sensor is likely the problem, the proper method is to measure its resistance. Based on symptoms from my experience, in such cases the error typically appears immediately after a reboot or within 30 to 60 min of operation.

Base 4: In my experience, most cases of this error are due to dirty heat exchangers. There is a gas heat exchanger and a plate one. Soot builds up in them over time. A faulty fan or turbine is also a possibility. Its bearings wear out, and it starts rotating worse. If you spin it by hand, it will feel stiffer than a new one and might have some play. It could also hum. You can replace it yourself, as they are sold in online stores. If you want, you can try to repair the old one, but they are typically not designed to be taken apart. In any case, always follow safety procedures. Turn off the gas and electricity supply before working on the boiler. Always use a gas detector. I don't recommend checking the fan draft with matches. Before removing the fan, avoid touching the pressure sensor; instead, disconnect its tubes. Before installing the new fan, remember to clean the exhaust housing and center it using the three bolts so that it does not touch the fan itself.

References

1. Luo, Z.; Yang, Z.; Xu, Z.; Yang, W.; Du, X. LLM4SR: A Survey on Large Language Models for Scientific Research. *arXiv* **2025**, arXiv:2501.04306. <https://doi.org/10.48550/arXiv.2501.04306>.
2. Busch, F.; Hoffmann, L.; Rueger, C.; van Dijk, E.H.; Kader, R.; Ortiz-Prado, E.; Makowski, M.R.; Saba, L.; Hadamitzky, M.; Kather, J.N.; et al. Current applications and challenges in large language models for patient care: A systematic review. *Commun. Med.* **2025**, *5*, 26. <https://doi.org/10.1038/s43856-024-00717-2>.
3. Salih, S.; Husain, O.; Hamdan, M.; Abdelsalam, S.; Elshafie, H.; Motwakel, A. Transforming education with AI: A systematic review of ChatGPT's role in learning, academic practices, and institutional adoption. *Results Eng.* **2025**, *25*, 103837. <https://doi.org/10.1016/j.rineng.2024.103837>.
4. Liu, O.; Fu, D.; Yogatama, D.; Neiswanger, W. DeLLMa: Decision Making Under Uncertainty with Large Language Models. *arXiv* **2024**, arXiv:2402.02392. <https://doi.org/10.48550/arXiv.2402.02392>.
5. Handler, A.; Larsen, K.R.; Hackathorn, R. Large language models present new questions for decision support. *Int. J. Inf. Manag.* **2024**, *79*, 102811. <https://doi.org/10.1016/j.ijinfomgt.2024.102811>.
6. Raza, M.; Jahangir, Z.; Riaz, M.B.; Saeed, M.J.; Sattar, M.A. Industrial applications of large language models. *Sci. Rep.* **2025**, *15*, 13755. <https://doi.org/10.1038/s41598-025-98483-1>.
7. Liu, X.; Erkoyuncu, J.A.; Fuh, J.Y.H.; Lu, W.F.; Li, B. Knowledge extraction for additive manufacturing process via named entity recognition with LLMs. *Robot. Comput.-Integr. Manuf.* **2025**, *93*, 102900. <https://doi.org/10.1016/j.rcim.2024.102900>.
8. Zakari, R.Y.; Owusu, J.W.; Qin, K.; Wang, H.; Lawal, Z.K.; He, T. VQA and Visual Reasoning: An overview of approaches, datasets, and future direction. *Neurocomputing* **2025**, *622*, 129345. <https://doi.org/10.1016/j.neucom.2025.129345>.

9. Spoladore, D.; Negri, L.; Arlati, S.; Mahroo, A.; Fossati, M.; Biffi, E.; Davalli, A.; Trombetta, A.; Sacco, M. Towards a knowledge-based decision support system to foster the return to work of wheelchair users. *Comput. Struct. Biotechnol. J.* **2024**, *24*, 374–392. <https://doi.org/10.1016/j.csbj.2024.05.013>.
10. Sheikhhoshkar, M.; El-Haouzi, H.B.; Aubry, A.; Hamzeh, F.; Rahimian, F. A data-driven and knowledge-based decision support system for optimized construction planning and control. *Autom. Constr.* **2025**, *173*, 106066. <https://doi.org/10.1016/j.autcon.2025.106066>.
11. Zhang, Y.; Chen, Y.; Li, X. Integrated framework of knowledge-based decision support system for user-centered residential design. *Expert Syst. Appl.* **2023**, *216*, 119412. <https://doi.org/10.1016/j.eswa.2022.119412>.
12. Psarommatis, F.; Kiritsis, D. A hybrid Decision Support System for automating decision making in the event of defects in the era of Zero Defect Manufacturing. *J. Ind. Inf. Integr.* **2022**, *26*, 100263. <https://doi.org/10.1016/j.jii.2021.100263>.
13. Wu, F.; Shen, T.; Bäck, T.; Chen, J.; Huang, G.; Jin, Y.; Kuang, K.; Li, M.; Lu, C.; Miao, J.; et al. Knowledge-Empowered, Collaborative, and Co-Evolving AI Models: The Post-LLM Roadmap. *Engineering* **2025**, *44*, 87–100. <https://doi.org/10.1016/j.eng.2024.12.008>.
14. Bettoni, M.; Obeng, E. Joint Knowledge Base: A Key to Knowledge Sharing and Collaboration. *Int. J. Manag. Sci. Bus. Adm.* **2023**, *9*, 36–45. <https://doi.org/10.18775/ijmsba.1849-5664-5419.2014.95.1003>.
15. Zhang, F.; Yuan, N.J.; Lian, D.; Xie, X.; Ma, W.-Y. Collaborative Knowledge Base Embedding for Recommender Systems. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 353–362. <https://doi.org/10.1145/2939672.2939673>.
16. Fuchs, M.; Beckert, F.; Biedermann, J.; Nagel, B. A collaborative knowledge-based method for the interactive development of cabin systems in virtual reality. *Comput. Ind.* **2022**, *136*, 103590. <https://doi.org/10.1016/j.compind.2021.103590>.
17. Nguyen, N.-K.; Nguyen, V.-H.; Le, A.-C. Formal Knowledge Augmented Language Models for Explainable and Robust Reasoning. *J. Intell. Fuzzy Syst.* **2025**, *50*, 2162–2176. <https://doi.org/10.1177/18758967251394597>.
18. Glukhikh, I.; Chernysheva, T.; Glukhikh, D. Neural network models for situation similarity assessment in hybrid-CBR. *J. Intell. Fuzzy Syst.* **2023**, *44*, 7669–7682. <https://doi.org/10.3233/JIFS-221335>.
19. Allen, B.P.; Stork, L.; Groth, P. Knowledge Engineering Using Large Language Models. *Trans. Graph Data Knowl.* **2023**, *1*, 3:1–3:19. <https://doi.org/10.4230/TGDK.1.1.3>.
20. Cheng, Q.; Chen, L.; Hu, Z.; Tang, J.; Xu, Q.; Ning, B. A novel prompting method for few-shot NER via LLMs. *Nat. Lang. Process. J.* **2024**, *8*, 100099. <https://doi.org/10.1016/j.nlp.2024.100099>.
21. Brand, F.; Malburg, L.; Bergmann, R. Large Language Models as Knowledge Engineers. In Proceedings of the ICCBR CBR-LLM'24: Workshop on Case-Based Reasoning and Large Language Model Synergies at ICCBR2024, Mérida, Mexico, 1 July 2024. Available online: https://ceur-ws.org/Vol-3708/paper_01.pdf (accessed on 13 February 2026).
22. Schinckus, M.; Simonofski, A.; Bono Rosselló, N. Large Language Models for Process Knowledge Acquisition. *Bus. Inf. Syst. Eng.* **2025**, *68*, 7–33. <https://doi.org/10.1007/s12599-025-00976-w>.
23. Shu, L.; Yeganeh, A.; Zhao, D. Large Language Models for Building Energy Retrofit Decision-Making: Technical and Sociotechnical Evaluations. *Buildings* **2025**, *15*, 4081. <https://doi.org/10.3390/buildings15224081>.
24. Haruehansapong, K.; Rongprom, W.; Kiangkhlaio, M.; Yeranee, K.; Sahoh, B. Deep Learning-Driven Automated Fault Detection and Diagnostics Based on a Contextual Environment: A Case Study of HVAC System. *Buildings* **2023**, *13*, 27. <https://doi.org/10.3390/buildings13010027>.
25. Jahanbakhsh, N.; Vega-Barbas, M.; Pau, I.; Elvira-Martín, L.; Moosavi, H.; García-Vázquez, C. Leveraging Retrieval-Augmented Generation for Automated Smart Home Orchestration. *Future Internet* **2025**, *17*, 198. <https://doi.org/10.3390/fi17050198>.
26. Karakurt, E.; Akbulut, A. Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs) for Enterprise Knowledge Management and Document Automation: A Systematic Literature Review. *Appl. Sci.* **2026**, *16*, 368. <https://doi.org/10.3390/app16010368>.
27. Xu, K.; Zhang, K.; Li, J.; Huang, W.; Wang, Y. CRP-RAG: A Retrieval-Augmented Generation Framework for Supporting Complex Logical Reasoning and Knowledge Planning. *Electronics* **2025**, *14*, 47. <https://doi.org/10.3390/electronics14010047>.
28. Xu, Y.; Guo, X.; Zeng, Z.; Miao, C. SoftCoT: Soft Chain-of-Thought for Efficient Reasoning with LLMs. *arXiv* **2025**, arXiv:2502.12134v1.
29. Xiao, Y.; Zhou, C.; Zhang, Q.; Dong, S.; Chen, S.; Huang, X. LAG: Logic-Augmented Generation from a Cartesian Perspective. *arXiv* **2025**, arXiv:2508.05509v1.

30. Gajderowicz, B.; Bhardwaj, A.; Fox, M. Convergence of Large-Language Models and Ontologies (ONTOLLM). In Proceedings of the Joint Ontology Workshops (JOWO)—Episode XI: The Sicilian Summer Under the Etna, Co-Located with the 15th International Conference on Formal Ontology in Information Systems (FOIS 2025), Catania, Italy, 8–9 September 2025. Available online: https://eil.mie.utoronto.ca/wp-content/uploads/2025/09/ONTOLLM_2025_Aug20v11.pdf (accessed on 13 March 2026).
31. Wiratunga, N.; Abeyratne, R.; Jayawardena, L.; Martin, K.; Massie, S.; Nkisi-Orji, I.; Weerasinghe, R.; Liret, A.; Fleisch, B. CBR-RAG: Case-Based Reasoning for Retrieval Augmented Generation in LLMs for Legal Question Answering. *arXiv* **2024**, arXiv:2404.04302v1.
32. Yang, R. CaseGPT: A Case Reasoning Framework Based on Language Models and Retrieval-Augmented Generation. *arXiv* **2024**, arXiv:2407.07913v1.
33. Hatalis, K.; Christou, D.; Kondapalli, V. Review of Case-Based Reasoning for LLM Agents: Theoretical Foundations, Architectural Components, and Cognitive Integration. *arXiv* **2025**, arXiv:2504.06943v2.
34. Watson, I. A Case-Based Persistent Memory for a Large Language Model. In Proceedings of the ICCBR CBR-LLM'24: Workshop on Case-Based Reasoning and Large Language Model Synergies at ICCBR2024, Merida, Mexico, 1 July 2024. Available online: https://ceur-ws.org/Vol-3708/paper_02.pdf (accessed on 13 March 2026).
35. Zheng, J.H.; Zhai, L.X.; Li, F.; Wang, D.; Li, Y.; Li, Z.; Wu, Q.H. Multi-objective optimization and multi-attribute decision-making support for optimal operation of multi stakeholder integrated energy systems. *Appl. Soft Comput.* **2024**, *167*, 112426. <https://doi.org/10.1016/j.asoc.2024.112426>.
36. Soori, M.; Karimi Ghaleh Jough, F.; Dastres, R.; Arezoo, B. AI-Based Decision Support Systems in Industry 4.0: A Review. *J. Econ. Technol.* **2026**, *4*, 206–225. <https://doi.org/10.1016/j.ject.2024.08.005>.
37. González-Villa, J.; Cuesta, A.; Spagnolo, M.; Zannotti, M.; Summers, L.; Elms, A.; Dhaya, A.; Jedlička, K.; Martolos, J.; Cetinkaya, D. Decision-support system for safety and security assessment and management in smart cities. *Multimed. Tools Appl.* **2024**, *83*, 61971–61994. <https://doi.org/10.1007/s11042-023-16020-6>.
38. Caro-Martínez, M.; Recio-García, J.A.; Díaz-Agudo, B.; Darias, J.M.; Wiratunga, N.; Martin, K.; Wijekoon, A.; Nkisi-Orji, I.; Corsar, D.; Pradeep, P.; et al. iSee: A case-based reasoning platform for the design of explanation experiences. *Knowl.-Based Syst.* **2024**, *302*, 112305. <https://doi.org/10.1016/j.knosys.2024.112305>.
39. Li, Y.; Du, H.; Kumaraswamy, S.B. Case-based reasoning approach for decision-making in building retrofit: A review. *Build. Environ.* **2024**, *248*, 111030. <https://doi.org/10.1016/j.buildenv.2023.111030>.
40. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv* **2023**, arXiv:2312.10997.
41. James, A.; Trovati, M.; Bolton, S. Retrieval-Augmented Generation to Generate Knowledge Assets and Creation of Action Drivers. *Appl. Sci.* **2025**, *15*, 6247. <https://doi.org/10.3390/app15116247>.
42. Tam, T.Y.C.; Sivarajkumar, S.; Kapoor, S.; Stolyar, A.V.; Polanska, K.; McCarthy, K.R.; Osterhoudt, H.; Wu, X.; Visweswaran, S.; Fu, S.; et al. A framework for human evaluation of large language models in healthcare derived from literature review. *NPJ Digit. Med.* **2024**, *7*, 258. <https://doi.org/10.1038/s41746-024-01258-7>.
43. Gupta, I.; Nagpal, G. *Artificial Intelligence and Expert Systems*; Mercury Learning & Information: Dulles, VA, USA, 2020; 300p.
44. Wu, C.; Ding, W.; Jin, Q.; Jiang, J.; Jiang, R.; Xiao, Q.; Liao, L.; Li, X. Retrieval augmented generation-driven information retrieval and question answering in construction management. *Adv. Eng. Inform.* **2025**, *65*, 103158. <https://doi.org/10.1016/j.aei.2025.103158>.
45. Kumar, E. *Knowledge Engineering*; TechSar Pvt. Ltd.: New Delhi, India, 2018; 392p.
46. Xiao, H.; Zhou, F.; Liu, X.; Liu, T.; Li, Z.; Liu, X.; Huang, X. A comprehensive survey of large language models and multimodal large language models in medicine. *Inf. Fusion* **2025**, *117*, 102888. <https://doi.org/10.1016/j.inffus.2024.102888>.
47. Krishnan, S.B.P. LLM Evaluations: A Survey of Programmatic, Human, and LLM-as-Judge Approaches. *Int. J. Innov. Res. Sci. Eng. Technol.* **2025**, *14*, 15108. <https://doi.org/10.15680/IJIRSET.2025.1406011>.
48. Kostopoulos, G.; Davrazos, G.; Kotsiantis, S. Explainable Artificial Intelligence-Based Decision Support Systems: A Recent Review. *Electronics* **2024**, *13*, 2842. <https://doi.org/10.3390/electronics13142842>.
49. Arslan, M.; Munawar, S.; Cruz, C. Sustainable Digitalization of Business with Multi-Agent RAG and LLM. *Procedia Comput. Sci.* **2024**, *246*, 4722–4731. <https://doi.org/10.1016/j.procs.2024.09.337>.
50. Holzinger, A.; Longo, L.; Cangelosi, A.; Ser, J.D. Research Frontiers in Machine Learning & Knowledge Extraction. *Mach. Learn. Knowl. Extr.* **2026**, *8*, 6. <https://doi.org/10.3390/make8010006>.
51. Taipalus, T. Vector database management systems: Fundamental concepts, use-cases, and current challenges. *Cogn. Syst. Res.* **2024**, *85*, 101216. <https://doi.org/10.1016/j.cogsys.2024.101216>.

52. Guo, Z.; Xia, L.; Yu, Y.; Ao, T.; Huang, C. Lightrag: Simple and fast retrieval-augmented generation. *arXiv* **2026**, arXiv:2410.05779v3.
53. Igolnikov, N.; Abbas, N. AtomRAG: Atomic query decomposition for multi-hop retrieval-augmented generation. *Acad. AI Appl.* **2026**, *2*, 1–9. <https://doi.org/10.20935/AcadAI8242>.
54. Teckwani, S.H.; Wong, A.H.-P.; Luke, N.V.; Low, I.C.C. Accuracy and reliability of large language models in assessing learning outcomes achievement across cognitive domains. *Adv. Physiol. Educ.* **2024**, *48*, 904–914.8 <https://doi.org/10.1152/advan.00137.2024>.
55. Tenakwah, E.S.; Boadu, G.; Tenakwah, E.J.; Parzakonis, M.; Brady, M.; Kansiime, P.; Said, S.; Eyaa, S.; Ayilu, R.K.; Radavoi, C.; et al. Assessment in the age of artificial intelligence: Interdisciplinary analysis of ChatGPT response to higher education assessment tasks. *Knowl. Manag. E-Learn.* **2025**, *17*, 1–48. <https://doi.org/10.34105/j.kmel.2025.17.001>.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.