POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria della Produzione Industriale e dell'Innovazione Tecnologica

Tesi di Laurea Magistrale

Verifiable and Human-Governed LLM Diagnosis for Industrial Predictive Maintenance: The AURA-PHM Framework

Relatori:
Prof. Alessandro Simeone
Prof. Li Yi
Prof. Gongzhuang Peng

Candidato:
Niccolò Pietro Rossi
Matricola: s337143

Anno Accademico 2025/2026

Abstract

Large language models can turn condition-monitoring data into readable fault diagnoses. In a safety-critical industrial setting, however, the problem is not fluency but trust. This thesis develops AURA-PHM, a diagnostic decision-support framework for wind-turbine maintenance that combines a Normal Behaviour Model, a retrieval-augmented reasoning layer and a human-oversight layer. Its central idea is that an LLM diagnosis can be relied on when its role is made verifiable and kept under human control, rather than when the model is made more accurate.

The framework rests on five contributions, each a facet of that idea. First, every diagnosis is traceable: it cites the specific sources and evidence that support it. Second, the knowledge base is stratified into two retrieval pools of distinct epistemic status, so that a citation can be placed as technical or operational. Third, operational urgency is set by a deterministic rule over a data-driven health index, computed outside the model, which keeps the critical decision out of the model's hands. Fourth, operator feedback is captured, indexed and carried across shifts, turning tacit operational knowledge into an explicit, retrievable form. Fifth, the evaluation is reported openly, quantitatively with RAGAS and qualitatively through end-to-end traces, with its biases stated rather than hidden.

The framework is validated on the open EDP wind-farm dataset and a documented gearbox failure. Its boundaries are stated plainly: fault detection is not a contribution, and the link from a residual to a failure mode is a physically motivated proposal, not a certified correspondence. The result is a diagnosis an operator can follow, question and carry forward.

Contents

Abstract 2

Contents 3

1. Introduction 8

1.1 Industrial context and the role of predictive maintenance 8

1.2 Predictive maintenance with Large Language Models: opportunities and trust challenges 10

1.3 Justification of research 11

1.4 Research objectives and contributions 12

1.5 Thesis structure 14

2. Background and Related Work: Foundations and Limitations of LLM-Assisted Industrial Diagnosis 16

2.1 Predictive Maintenance and SCADA-Based Anomaly Detection in Wind Turbines 16

2.2 Industrial Standards in Maintenance 18

2.3 LLMs in Industrial Diagnostics 21

2.4 Retrieval-Augmented Generation in Technical Domains 24

2.5 LLM Explainability and Trust in Critical Systems 27

2.6 Evaluating LLM-Based Diagnostic Systems: RAGAS and the Self-Preference Bias 30

2.7 Operational Continuity and Shift Handover in Industrial Maintenance 33

2.8 Research Gaps and Positioning of the Present Thesis 36

3. The AURA-PHM Framework: A Layered Architecture for Traceable Diagnostic Decision Support 41

3.1 Architectural overview and design principles 41

3.2 The ingestion and detection layer 44

3.2.1 Data ingestion and conditioning 45

3.2.2 The Normal Behaviour Model 46

3.2.3 Health assessment and alarm generation 47
3.3 The diagnostic reasoning layer 48
3.3.1 The stratified knowledge base 48
3.3.2 The diagnostic reasoning engine 49
3.4 The human oversight and persistence layer 51
3.4.1 Traceability and audit 51
3.4.2 Operator interaction and shift continuity 52
3.5 Information flow and inter-layer interfaces 54
4. Instantiation on the Wind Turbines Validation Case 58
4.1 From framework to running system 58
4.2 Instantiating the ingestion and detection layer 61
4.2.1 Data ingestion and conditioning 61
4.2.2 Training and the Normal Behaviour Model 62
4.2.3 The Health Index and its thresholds 63
4.3 Populating the stratified knowledge base 64
4.3.1 The four technical strata 65
4.3.2 Segmentation embedding and pool construction 66
4.3.3 The runtime operational memory 67
4.4 Instantiating the diagnostic reasoning layer 68
4.4.1 The semantic query classifier 69
4.4.2 Pool routing and retrieval 69
4.4.3 Constrained generation and determinist post processing 70
4.5 Instantiating the human oversight and persistence layer 72
4.5.1 The audit trail 72
4.5.2 The operator feedback mechanism 74
4.5.3 The shift briefing and cross-shift continuity 75
4.6 Persistent state and reconstruction 75

5. Experimental Validation 78

5.1 Experimental setup and evaluation protocol 78

5.2 Detection layer on the 2017 stream 79

5.3 Quantitative evaluation of the diagnostic layer 81

5.4 Qualitative validation of the diagnostic layer 83

5.4.1 Query classification and pool routing 83

5.4.2 End-to-end diagnostic traces 84

5.5 Cross-shift knowledge continuity 87

5.6 Discussion 89

5.6.1 Traceability and epistemic separation 89

5.6.2 Cross-shift continuity as a testbed of traceability 90

5.6.3 Evaluating the system honestly 91

5.7 Real-world grounding, validity and limitations 92

6. Conclusions 95

6.1 Summary 95

6.2 Contributions 95

6.3 Limitations and future directions 96

6.4 Concluding remarks 97

References 99

Appendices 103

Appendix A: The stratified knowledge base: source documents 103

A.1: DOC1 Gearbox FMEA and failure-mode taxonomy 103

A.2: DOC2 EDP fault archive (empirical stratum) 107

A.3: DOC3 Operational context of the EDP fleet 109

A.4: DOC4 Urgency and action guide 110

Appendix B: Prompts and generation configuration 112

B.1: Diagnostic system prompt and user template 112

B.2: Shift-briefing prompt 114

Appendix C: Evaluation set and per-question RAGAS results 114

Appendix D: Full diagnostic and classification traces 115

Appendix E: Audit trail and shift-continuity artefacts 118

E.2: Sample audit-trail rows 118

E.3: Example generated shift briefing (8-9 October 2017) 118

1.Introduction

This chapter frames the problem and states what the thesis adds. It opens with the industrial context: why predictive maintenance sits where it does in modern asset management, and the precise point in the diagnostic workflow where Large Language Models have started to appear. That point is also where trust becomes an issue, at least in settings where a wrong call carries safety consequences. The chapter then names the gaps that current LLM-assisted diagnostic systems leave open, sets the objectives and contributions against them, and ends with a map of the document.

1.1 Industrial context and the role of predictive maintenance

Industry 4.0, the label given to the fourth industrial revolution, reshaped how connected assets are managed, how maintenance is done, and how operational decisions get made across manufacturing and process industries. Its ingredients are familiar: smart sensors, big data, artificial intelligence, the Internet of Things. Out of that convergence, predictive maintenance moved to the centre of industrial strategy [1]. The reason is not subtle. In a modern plant a single breakdown spreads its effects across safety, output, energy use, and cost, and often does so in ways that are hard to undo once the failure has happened [1].

Maintenance strategy has shifted a good deal over three decades, and the trajectory mirrors that same rise in importance. Early practice was mostly corrective: fix it after it breaks. Through the 1990s and 2000s, preventive and condition-based approaches took hold, backed by dedicated computerized maintenance management systems [2]. Then the 2010s changed the picture, on the back of Industry 4.0 technologies, machine-learning-driven prediction, high-resolution condition monitoring, cyber-physical systems and digital twins [2]. The most recent step is prescriptive maintenance, where the system is expected not just to forecast the health of a machine but to say which action to take [2].

Inside the predictive-maintenance workflow, data-driven detectors, anomaly models and Normal Behaviour Models above all, do one thing: they flag that the asset's state has moved. That flag starts the decision, it does not settle it. Detection raises the next round of questions. What is the likely physical cause? How fast will the situation develop? What should be done, and how urgently? And who needs to be told, on what evidence?

That second phase, turning an anomaly into a diagnosis that can be acted on, audited, and reused, is what this thesis is about. The detection layer itself is not the target; it rests on established Normal Behaviour Modelling and stays as it is. The work sits above it, in a

diagnostic and decision-support layer that traces its own reasoning, ties every claim to a checkable reference, and, over time, absorbs what operators know.

The stakes of good diagnostic decision support are real in any plant. In critical infrastructure they sharpen. Unplanned downtime there costs out of all proportion, the windows for intervention are limited by sheer physical access, and a degradation missed can end in a safety event rather than a repair bill. Wind energy is a good place to test a diagnostic architecture against exactly these pressures, and it is the domain this thesis uses for validation. Four properties of the sector, converging, explain the choice.

First, the data exists in the open. SCADA datasets of real scale and duration are publicly available, which gives systematic experimentation something solid to stand on.

Second, turbine maintenance answers to recognised international standards, IEC 61400-4 for gearbox design and operation, ISO 20816-21 for vibration measurement. These give the diagnostic knowledge base an auditable frame of reference.

Third, the Prognostics and Health Management literature for wind turbines is mature and well documented, so the work here can be placed against baselines that already exist.

Fourth, turbines run at remote sites, cost a great deal when they fail, and are looked after by operators working in shifts. Under those conditions, auditable diagnoses and knowledge that survives a shift handover stop being abstract virtues. They become daily requirements.

Take the economics first. Operation and maintenance can eat up to 30% of the Levelised Cost of Energy for offshore installations, and that share is set to climb as new farms move further from shore [3]. Unscheduled work is what drives the bill. A failure nobody saw coming means an expensive repair, technicians mobilised, and downtime that weather or sea-state can stretch out for days. The gearbox shows the pattern at its sharpest. Carroll et al., looking at roughly 350 offshore turbines across Europe, put the average failure rate at about 8.3 failures per turbine per year across the monitored fleet [4]. The gearbox sits among the costliest of those: a major replacement averages €230,000, and repair times run past 230 hours, longer than any other subassembly on the machine [4]. Frequent enough to matter, expensive to fix, slow to recover, gearbox diagnostics make a high-impact target for validating the architecture proposed here.

1.2 Predictive maintenance with Large Language Models: opportunities and trust challenges

The diagnostic phase, in the workflow just sketched, opens once a detection model has raised an anomaly. In most established pipelines that signal is a number: a residual past a learned threshold, an index dropping below a calibrated level, an alarm from a Normal Behaviour Model. The Normal Behaviour Model approach has proven itself across many industrial settings, wind energy among them [5], [6]. For gearboxes specifically, several studies have caught degradation weeks or months before a catastrophic failure [5], [7], [8].

A number, though, is not a decision, whatever the industry. Tell a technician that the residual on a measured temperature has run twelve degrees above expectation and you have handed them a figure, not an answer. What they actually need is a diagnosis: what is probably going wrong, how it will develop, what to do, and how soon. For years that leap, from anomaly to something you can act on, rested on the tacit experience of senior engineers. Which meant the quality of the response turned on who happened to be on shift that day.

Large Language Models have opened a new way to close that gap. Given structured context, recent models reason well, and a number of studies have already put them to work on industrial fault diagnostics, machinery health monitoring, and decision support for energy systems [9], [10], [11], [12]. The appeal, here, is specific: an LLM can take a raw anomaly signal and turn it into a natural-language diagnosis grounded in technical documentation, one that cites standards, past cases, and operating guidelines. In other words, the kind of operable reading that used to live only in an operator's head.

That appeal comes with a matching problem. LLMs hallucinate: they produce fluent output that no factual evidence supports [13], [14], [15]. Somewhere trivial, a hallucinated answer is a nuisance. In critical infrastructure, where a diagnosis can decide whether a machine is shut down, whether a crew is sent out, or whether a maintenance plan worth tens of thousands of Euros gets signed off, the same slip becomes a safety and economic risk. Mirshekali et al. put it plainly, noting that "LLM decision processes are opaque, posing challenges in environments that require traceability and justification" and that this missing explainability "limits trust and regulatory acceptance, especially in critical infrastructure settings" [13]. The rest of the recent literature says much the same, with author after author asking for explicit knowledge grounding, verification frameworks, and structured traceability [14], [15], [16].

None of this is an academic worry. It is a concrete obstacle to putting these systems to work. A technician who cannot check where an LLM diagnosis came from cannot, in good professional conscience, act on it. A regulator who cannot audit how a decision was reached cannot certify the system. A maintenance manager who cannot say why an automated recommendation was followed has nothing to stand on if the decision is later disputed. **Without traceability, grounding, and human oversight, an LLM-based diagnostic system cannot be responsibly deployed as a decision-support tool in industrial settings.**

1.3 Justification of research

The literature has three main answers to the trust problem. Ground the model on a knowledge graph that constrains its reasoning paths [16], [17]. Wrap it in Retrieval-Augmented Generation so every response is anchored in retrieved documentation [14], [15], [18]. Or keep a human in the loop, with an operator confirming or correcting the output before anyone uses it [19], [20]. Each has merit, and this work leans on all three. Chapter 2 takes them apart in detail; what matters for now is that each one, as the current literature applies it, leaves something open. Three of those gaps drive this thesis.

The first gap is the knowledge base itself. Existing systems tend to pour paper extracts, maintenance manuals, and historical reports into one retrieval space, with no explicit line between sources of different epistemic status. That hurts twice. For an auditor, there is no easy way to tell whether a cited fragment came from a stable technical reference or from a dynamic operational note. For day-to-day use, mixing long-term technical knowledge with short-term operational knowledge makes the base slower and messier to maintain. Split those sources into distinct retrievable pools and both problems ease, and the workflow becomes easier to see through.

The second gap is traceability, which is rarely treated as a property the system must guarantee. Plenty of LLM-based predictive-maintenance frameworks report aggregate performance, yet offer no structured way to pull back, for one specific diagnosis, the documentary fragments that produced it. So even a grounded system can stay opaque in practice. The operator sees a citation in the generated text but cannot always check it then and there, inside the live workflow. Trust suffers, external audit gets harder, and the process is less reliable than its numbers suggest.

The third gap is continuity across shifts, oddly underexplored in the LLM-PHM literature given how well known the weakness is on the shop floor. The operator who spots an anomaly is often not the one handling the case hours later, and what passes between them tends to be tacit rather than written down. Observations, hunches, contextual judgements made on one shift slip away, or survive only in an informal note. The system never gets to build on the experience accumulating across shifts.

AURA-PHM is the response to these three gaps: a traceable RAG-LLM architecture for diagnostic decision support in industrial predictive maintenance. It is built around a stratified knowledge base, gives every diagnostic output explicit traceability, and keeps capturing and re-indexing what operators write, shift after shift. The validation runs on a wind turbine gearbox case, but cross-domain applicability was a design goal from the start, not an afterthought.

1.4 Research objectives and contributions

This thesis sets out to develop, implement, and empirically validate a diagnostic decision-support architecture for industrial predictive maintenance, one whose outputs are traceable, anchored to a stratified and auditable knowledge base, and enriched by continuous operator feedback. Three research objectives carry that aim. They are stated in parallel, each answering one of the limitations laid out in *Section 1.3* and developed in *Chapter 2*.

Objective 1: Build a Retrieval-Augmented Generation architecture for industrial fault diagnosis in which every generated output can be traced back to the specific sources behind it, so that operators and external auditors can verify it after the fact.

Objective 2: Construct a stratified knowledge base that separates sources of different epistemic status into distinct retrievable pools, and propose a methodological bridge between data-driven anomaly indicators and the qualitative failure-mode taxonomy set out in certified industrial standards.

Objective 3: Show, through end-to-end testing on a real industrial dataset, that observations an operator enters during a shift can be captured, structured, indexed, and made retrievable within the diagnostic knowledge base on later shifts, turning tacit operational knowledge into an explicit, retrievable form inside the LLM workflow.

Five principal contributions follow from this.

Contribution 1: A complete, working end-to-end prototype of AURA-PHM, coupling a Normal Behaviour Model detection layer to a RAG-LLM diagnostic layer, validated on the open-access Énergias de Portugal (EDP) SCADA dataset and against a documented gearbox failure from October 2017.

Contribution 2: A stratified knowledge base in which a stable technical pool, filled here from certified industrial standards, sits alongside a dynamic operational pool that grows as operators interact with the system. The split is structural rather than tied to any particular content: swap in proprietary technical documentation for a production deployment and the same auditability holds.

Contribution 3: A deterministic rule that turns the Health Index into a structured urgency label, fixing a known weakness in LLM-generated output and making the deterministic and LLM-chosen urgencies traceable side by side.

Contribution 4: An operator-feedback mechanism that records free-text notes alongside structured CONFIRMED/CORRECTED/PENDING verdicts, stores them in three independent channels, and brings them back in later sessions through the same semantic retrieval used for the technical pool, a concrete instance of tacit-to-explicit knowledge conversion inside the diagnostic workflow.

Contribution 5: A quantitative evaluation of the diagnostic component with the RAGAS framework, set against qualitative validation through four end-to-end use cases, one of them shift-to-shift knowledge continuity.

The validation stays on wind turbine gearboxes, but the methodological contributions, the stratified knowledge base, the traceable RAG diagnostic layer, the deterministic urgency mapping, the tacit-knowledge capture, were designed to travel. Whether they actually transfer to other industrial settings is left as future work, taken up in Chapter 6. Wind turbines earn their place as the test bed for the four reasons given in Section 1.1: open SCADA data, regulation by recognised standards, a mature PHM literature, and the plain operational weight of auditability and shift continuity at remote sites.

Objectives, gaps, contributions, and the chapters that develop them are drawn together in *Table 1.1*.

***Table 1.1**. Research objectives, the limitation each addresses, principal contributions, and chapters in which they are developed.*

Research objective	Limitation addressed	Principal contribution(s)	Chapter(s)
O1. *Develop a Retrieval-Augmented Generation architecture in which every diagnostic output can be traced back to the specific knowledge sources that supported it.*	Traceability of diagnostic output	**C1.** End-to-end AURA-PHM prototype (NBM detection layer + traceable RAG layer). **C5.** RAGAS evaluation and qualitative validation through end-to-end use cases.	Ch. 3, 4, 5
O2. *Construct a stratified knowledge base separating sources of different epistemic status into distinct retrievable pools, and bridge data-driven anomaly indicators with certified failure-mode taxonomies.*	Knowledge-base stratification	**C2.** Stratified KB architecture (certified-standards technical pool + dynamic operational pool; compatible with proprietary documentation). **C3.** Deterministic Health-Index-to-urgency mapping rule.	Ch. 3, 4
O3. *Demonstrate that operator-generated observations entered during a shift can be captured, structured, indexed, and made retrievable in subsequent shifts.*	Shift-to-shift continuity	**C4.** Operator-feedback mechanism with three persistence channels, instantiating tacit-to-explicit knowledge conversion.	Ch. 3, 4, 5

1.5 Thesis structure

What follows is organised in three blocks, matching the three logical phases of the contribution: foundation, development, validation.

The foundation block sets the conceptual and methodological ground. Chapter 2 reviews the state of the art across the four fields the architecture draws on: predictive maintenance and SCADA-based anomaly detection for industrial assets, wind turbines in particular; the use of certified industrial standards inside fault-diagnostic systems; Large Language Models and Retrieval-Augmented Generation in technical domains, the RAGAS evaluation framework included; and tacit knowledge management, read against shift-based industrial work. It closes by returning to the three gaps from Section 1.3 and sharpening them, against the reviewed literature, into five interconnected gaps that position the thesis as a response.

The development block lays out the architecture and puts it to work on the validation case. Chapter 3 presents AURA-PHM as three cooperating layers, ingestion and detection, diagnostic reasoning, and human oversight and persistence, each broken into its own components, described by role, by input-output interface, and by the engineering choices that shaped it. Chapter 4 grounds the framework on the wind turbine case: the Normal

Behaviour Model training, the stratified knowledge base built with explicit reference to certified standards, the semantic query classifier and the deterministic Health-Index-to-urgency rule, and the operator-feedback channels.

The validation block reports the results, weighs what they mean, and closes the thesis. **Chapter 5** presents the experimental validation on the EDP dataset along two axes: a quantitative RAGAS evaluation of the diagnostic component, and a qualitative one built from end-to-end traces, the semantic classifier routing operator questions to the right pool, the pipeline reasoning over real anomalies, and a single operator note carried from one shift into the next. It then reads those results against the five gaps consolidated in Chapter 2, setting the RAGAS scores beside the design choices and probing how faithful the generation stays at the seam between data-driven and standards-anchored content, before stating the limits of both the evaluation and the prototype. Chapter 6 ends the thesis, drawing the contributions together, pairing each limitation with the future work that would answer it, cross-domain transfer among them as flagged in Section 1.4, and reflecting on what a trustworthy LLM-assisted diagnosis actually demands.

2. Background and Related Work: Foundations and Limitations of LLM-Assisted Industrial Diagnosis

This chapter reviews the research the thesis builds on and positions itself against. It works through the domains the contribution pulls together: how mature SCADA-based anomaly detection has become in wind turbines, and the standards that govern the field; the use of Large Language Models in industrial diagnostics, together with the retrieval-augmented and explainability techniques meant to make them trustworthy; the evaluation of such systems, and the bias that muddies it; and a question most of the literature skips, continuity across shifts. It ends by pulling these threads into the specific gaps this work sets out to close.

2.1 Predictive Maintenance and SCADA-Based Anomaly Detection in Wind Turbines

The move from reactive and time-based maintenance toward predictive maintenance (PdM) has been one of the defining threads of industrial maintenance research over the past three decades [1], [2]. Wind energy, within that trajectory, turned into an unusually mature application domain. Two things pushed it there: the high operation and maintenance costs of running a wind farm, and the comparatively rich data infrastructure already sitting on most modern turbines.

Operation and maintenance can run to 30% of the total levelised cost of energy (LCOE) for an offshore farm [3], [4]. Most of that cost gathers in a handful of failure-prone subassemblies. Carroll et al. [4] studied roughly 350 offshore turbines across Europe and found the average failure rate settling at about 10 events per turbine per year by the third operational year, with the pitch and hydraulic system, the other-components group, and the generator leading the count. The gearbox is a different story: not the most frequent failure, but the most expensive one. A major replacement averages €230,000 per event, the highest per-failure cost of any subassembly, and about 231 hours of repair time, with full downtime stretching from days to weeks once travel, lead time, and inaccessibility are counted in. A follow-up study on gearboxes specifically [5] used thousands of labelled failure examples to predict failure and remaining useful life with machine learning, showing SCADA-based prediction up to a month ahead and vibration-based prediction five to six months ahead. Rare, then, but disproportionately costly, which is exactly why predictive strategies keep being argued for over reactive ones.

Anticipating failure takes data, and modern turbines carry two complementary streams of it. Condition Monitoring Systems (CMSs) pull high-frequency vibration signals from dedicated accelerometers, usually on the drivetrain and main bearings, and read mechanical faults through spectral analysis [6]. Supervisory Control and Data Acquisition (SCADA) systems work at the other end: lower-frequency operational and environmental signals, typically 10-minute averages of wind speed, power output, rotational speed, component temperatures, and a broad set of control variables, all of it available on every turbine in a farm at no extra instrumentation cost [6], [23]. That cost advantage is what has drawn so much research toward wringing condition information out of SCADA signals, the well-known limits of their temporal resolution notwithstanding [24]. This thesis works in that SCADA-based stream. It does not use vibration signals.

Inside SCADA-based condition monitoring, one paradigm dominates: the Normal Behaviour Model (NBM) [6], [23]. The premise is simple and does a lot of work. Train a data-driven model on healthy operational data to predict the expected value of a target sensor, usually a temperature tied to a critical component, from relevant inputs such as ambient temperature, generated power, and rotational speed. In service, the model keeps comparing its prediction against the live measurement, and a deviation past a set threshold reads as an early sign of degradation. The formulation needs no labelled fault data to train, which matters, since fault events are inherently scarce in operational datasets [25].

The predictor itself has been built many ways, broadly statistical methods, tree-based ensembles, and deep learning [6]. Recurrent, convolutional, and autoencoder networks have all seen wide use [6], [23], while tree-based methods like Random Forest and Gradient Boosting hold their own at lower computational cost and with clearer feature-importance readings. Puruncajas et al. [23] put an ANN-based NBM to work on early detection of gearbox bearing temperature anomalies across a multi-turbine SCADA dataset, catching incipient failures several months out from standard SCADA signals alone. Castellani et al. [7] showed, on a real multi-megawatt turbine, that folding vibration data into SCADA through a cascade of machine learning and deep learning models can stretch the diagnostic lead time, though only by paying for the extra CMS instrumentation the SCADA-only paradigm sets out to avoid.

A parallel line of work leaves the predictor alone and concentrates on the residual stage, where model output becomes an anomaly indication. The options run from fixed thresholds on training RMSE to statistical measures like the Mahalanobis distance, from change-point

detection to filtering schemes built to keep fault-related information alive through SCADA preprocessing [6], [25]. The sheer variety here says something about the field. The same NBM residual can be read quite differently depending on the post-processing logic laid over it, and what counts as a “true” anomaly is seldom defined apart from the detection pipeline that produced it.

For all its maturity, the SCADA-based PdM literature carries a few recurring blind spots, and they are part of what motivates this work. Detection has taken most of the attention, the moment of spotting that something is off, while the diagnostic stage that ought to follow gets far less. Once a residual clears its threshold, operators are usually left holding a number, with little contextual help to say what the anomaly means physically, which failure mode it points to, or what action it calls for [6], [10]. NBM outputs, too, are often judged in isolation against labelled fault events, with no account of how detection feeds the wider workflow of a maintenance team. And the comparative review by Pandit and Wang [6] notes that the field still has no agreed benchmark datasets, evaluation protocols, or reporting standards, which makes studies hard to compare and slows the discipline’s cumulative progress.

This is where the thesis takes its position. The Normal Behaviour Model paradigm, and the Random Forest formulation used here in particular, is treated as a mature, well-established detection layer to build a diagnostic decision-support system on top of. The contribution is not in tuning the NBM. It is in what happens once the NBM has done its job: how the residual signal becomes an operable, traceable, reusable diagnosis. A diagnostic layer of that kind has to be anchored in the technical reference frame of the wind sector, which is partly written down in a stable body of international standards. Quite how that codification can be made to work inside an automated diagnostic system is an open question in itself, and it sets up the next subsection.

2.2 Industrial Standards in Maintenance

The diagnostic decision-support layer proposed here cannot be defined in a normative vacuum. Wind turbine operations answer to a stable, internationally adopted body of standards that sets out design, condition monitoring, and reliability practice across the sector. Three matter for this work: two that bear on it directly, and a third that speaks to the probabilistic side of reliability assessment.

The first is **IEC 61400-4:2025**, the second edition of *Wind Energy Generation Systems, Part 4: Design Requirements for Wind Turbine Gearboxes*, published in May 2025 and superseding the 2012 edition [26]. It applies to enclosed speed-increasing gearboxes for horizontal-axis turbines rated above 500 kW, onshore and offshore alike. Its formal scope is design rather than maintenance, yet the content speaks directly to diagnostic systems, for two reasons. First, the standard requires a systematic critical-systems analysis, an FMEA or an FMECA, to run throughout the design process, identifying failure modes, their consequences, and their criticality [26]. That makes FMEA/FMECA a recognised formal reference for failure-mode classification within the wind-gearbox design lifecycle, a shared technical vocabulary a diagnostic system can adopt as its classification reference. Second, Table 10 sorts failure modes into four classes: A1 (deterministic load profile, validated calculation models available), A2 (deterministic load profile, validated models not available), B (stochastic load profile from friction, abrasion or environmental factors, some models available), and C (stochastic load profile from random impacts and other irregular phenomena, no validated models available) [26]. The standard's own examples include tooth bending and bearing rolling contact fatigue (A1), tooth flank fracture and bearing white etching cracks (A2), scuffing (B), and debris damage and fluting (C). For a system trying to map detected anomalies onto an established failure-mode vocabulary, this taxonomy is a natural reference frame.

The standard also concedes, in Section 4.3, that *"condition monitoring systems can further increase the availability as they can detect failures in due time to plan for repair"* [26]. Condition monitoring, in other words, is positioned as a complement to design-stage reliability assessment. But the standard sets no diagnostic thresholds, no alarm logic, no interpretation procedure. The failure-mode taxonomy is qualitative; the quantitative decision rules are left, explicitly, to the implementer.

The second is **ISO 20816-21:2025**, the first edition of *Mechanical Vibration, Measurement and Evaluation of Machine Vibration, Part 21: Horizontal Axis Wind Turbines*, published in May 2025 and superseding ISO 10816-21:2015 [27]. It covers horizontal-axis turbines with rated generator output above 200 kW, taking in both Group 1 (gearbox-coupled) and Group 2 (direct-drive) machines. ISO 20816-21 standardises how vibration is measured, recommends evaluation zones for continuous-load operation, and sets assessment quantities, acceleration and velocity over specified frequency bands, that let machines be compared against one another. Annex A, drawn from vibration data on thousands of turbines rated at 5

MW or less, gives evaluation-zone boundaries meant to help operators and manufacturers open early-fault discussions, though the standard is candid that *"in most cases these evaluation zone boundaries are not suitable for the early detection of faults"* [27].

One point matters especially here: the standard's own limit on its scope. ISO 20816-21 states outright that *"this document does not address diagnosis or fault detection, although the measurement equipment described can be used for vibration monitoring"* [27]. It fixes the measurement reference frame for condition monitoring, then stops short of any diagnostic logic. Diagnosis is left as a downstream activity the standard supports without prescribing. The vibration-based procedures of ISO 20816-21 fall outside this thesis's SCADA-only data scope, but its clean line between measurement and diagnosis is methodologically useful whatever the data modality.

A third, relevant to the wider reliability context, is **IEC TS 61400-9:2025**, the first edition of *Wind Energy Generation Systems, Part 9: Probabilistic Design Measures for Wind Turbines*, published in July 2025 [28]. This Technical Specification brings in probabilistic methods as an alternative to the semi-probabilistic, deterministic approaches of IEC 61400-1, and calls for Failure Mode and Effects Analysis, or an equivalent, in the design of control and protection systems. Its interest here runs past design. The probabilistic philosophy of IEC TS 61400-9 rhymes with the health-index reasoning used in this thesis, where a component's health is a continuous degree of deviation from normal behaviour rather than a binary fault flag. So the standard offers not just a normative reference but a methodological backdrop that fits the diagnostic approach developed later.

Put together, the three standards mark out the normative frame any wind turbine diagnostic system works inside. IEC 61400-4 supplies the failure-mode taxonomy. ISO 20816-21 supplies the measurement framework for condition monitoring. IEC TS 61400-9 grounds the reliability methodology in probabilistic terms. Their shared trait, though, is also their shared limitation once you look from an operational diagnostic system's point of view. Not one of them prescribes an operational diagnostic logic layer. The translation of measurements, classifications, and reliability assessments into real-time interpretive decisions is left to whoever implements it.

That gap sets the thesis's stance toward the standards. AURA-PHM does not try to replace or extend the normative frame; it builds on it. The IEC 61400-4 failure-mode taxonomy becomes the structured vocabulary that diagnostic outputs are mapped against; the

measurement principles of ISO 20816-21 stand as the reference for what a measurable condition state even is. What the thesis adds is the layer joining that normative scaffolding to operational diagnostic decisions, in a form that can be traced and reused. There is a complication. The emerging family of Large Language Models puts a new strain on this scaffolding: their generative nature lets them phrase diagnoses in natural language, but also leaves them open to hallucination and to attribution drift between sources, neither of which the qualitative scaffolding of the standards can settle on its own. How that strain has been handled so far is the subject of the next subsection.

2.3 LLMs in Industrial Diagnostics

Sections 2.1 and 2.2 leave us with two matching observations. SCADA-based anomaly detection in wind turbines has reached methodological maturity, yet stops at the edge of operational diagnosis; the international standards fix a failure-mode vocabulary and a measurement reference frame, yet prescribe no operational logic for turning a detected anomaly into an interpretive decision. Over the last three years, the space between detection and diagnostic action has pulled in a different research community, the one working on Large Language Models (LLMs) for industrial fault diagnosis.

LLMs came to machinery health recently, and the field is widening fast. A PRISMA-guided systematic review by Tsallis et al. [12] went through 58 Scopus studies published between 2022 and early 2026, tracing the shift from text-centric LLMs toward autonomous, cyber-physical industrial agents. Among the primary results it gathers: multimodal foundation models pushing out-of-distribution diagnostic accuracy to 71.95%, against 18.25% for conventional models, and LLM systems paired with knowledge graphs reaching roughly 98% fault-diagnosis accuracy on selected benchmarks from just 1.2% labelled samples [12]. A caveat worth keeping in view. These are figures from individual studies the review collects, not measurements of the technology at large, and the review itself flags that real adoption will need targeted physics-AI hybridization, with real-time determinism, corpus poisoning, and edge deployment still unresolved.

Inside this fast-moving field, a few lines of work bear directly on the diagnostic problem this thesis takes on. One ties LLMs to knowledge graphs and retrieval so the reasoning stays anchored to verifiable industrial knowledge. Nie et al. [16] build a CNC machine-tool fault-diagnosis framework that joins a fault knowledge graph, assembled from maintenance

records, failure cases, and equipment manuals, to a fine-tuned GPT-4o through Retrieval-Augmented Generation (RAG), and report clear gains in fault-mode matching accuracy and diagnostic efficiency over traditional methods in a real workshop. Mirshekali et al. [13], reviewing LLMs for energy systems more broadly, catalogue applications across HVAC, building energy management, and grid monitoring, and pull out recurring themes, Shapley-based interpretability, schema-constrained outputs, physics-based validation, that line up with what safety-critical systems demand of trust. The same review is blunt that hallucination and brittleness stay open for industrial use, especially under noise or out-of-distribution inputs.

A second line uses LLMs to read sensor data directly. Qaid et al. [11] propose FD-LLM, which lines vibration signals up with LLM inputs by encoding them as text, so the model can do both classification-style fault diagnosis and context-aware spectrum language modelling. Tested across four open-source LLMs, it reports strong diagnostic performance, zero-shot adaptability across operating conditions, and solid generalisation in cross-dataset settings. That same work pairs LLM-as-a-Judge plausibility scoring with RAGAS faithfulness metrics, a combination revisited in Section 2.6. Encoding numbers as text is a real methodological move, though it also slips in a domain-translation step whose information loss often goes unmeasured.

A third line lands squarely on wind turbines. Tan and Carroll [3] set GPT-4o against DeepSeek-V3 for generating ML pipelines that classify gearbox failures from labelled SCADA data. Prompted zero-shot, DeepSeek-V3 produced a pipeline that beat both GPT-4o and fine-tuned Random Forest and XGBoost baselines on analytical scope and prediction accuracy [3]. The result matters twice over. It shows LLMs can act as capable generators of diagnostic ML pipelines for turbine SCADA, not only as direct classifiers. And it owns a basic limitation of off-the-shelf LLMs: identical prompts do not return perfectly identical answers, and developer-side optimisations shape response quality in ways the user cannot see [3]. For any system that wants to be traceable and reusable, that is a pointed problem.

A fourth line, and an increasingly visible one, casts LLMs not as classifiers or pipeline generators but as agents inside semi-autonomous maintenance workflows. Carvalho et al. [10], in a deployment-minded position paper, describe LLM-powered agents wired to time-series anomaly detection, edge-to-cloud orchestration, and few-shot learning, built to react to emerging failure modes in real time. They insist that operational trust and explainability, *“transparent, interpretable decision-making as AI agents operate autonomously in mission-*

critical settings" [10], are non-negotiable for industrial deployment, and flag open problems in latency, reliability, and the handling of autonomous decisions. Casting LLMs as agents inside a structured operational loop, rather than as standalone reasoners, speaks directly to the architecture developed in later chapters.

A fifth direction cuts across the rest and puts the weight on human-in-the-loop integration. The Tsallis et al. review [12] records that, across the studies examined, autonomous agentic architectures inside digital twins can cut false-positive alarms by as much as 67%, and that folding in retrieval-augmented generation and knowledge graphs blunts hallucination and corpus poisoning. It also treats human-machine collaboration, through intelligent interfaces and collaborative learning, as essential to safety-critical approvals, keeping accountability shared and staying compliant with industrial safety standards [12]. The pattern that keeps returning is this: LLMs work best hemmed in by structure, schemas, knowledge bases, validation loops, human oversight, rather than let loose as unconstrained generators. Many of the surveyed systems, it should be said, sit at prototype or proof-of-concept stage, and whether they scale to full production is not yet settled.

For all the momentum, the literature has limits that shape this thesis's contribution. Most LLM-based diagnostic systems are tested on idealised or proprietary benchmarks, with little validation on production-scale SCADA from working wind farms. The Tan and Carroll study [3] is one of the few that touches labelled turbine SCADA directly, and even there the evaluation stays binary rather than reaching multi-failure-mode diagnostic reasoning. Then there is the integration with codified industrial standards, the failure-mode taxonomies, design specifications, and condition-monitoring frameworks from Section 2.2, which stays largely informal. Standards rarely feature as a first-class part of the knowledge base, so the diagnostic output cannot easily be traced back to a specific normative reference. And the operational side, how a diagnosis meets shift workflows, operator handovers, and the day-to-day of a maintenance team, is acknowledged in the agent-oriented work [10], [12] but seldom built out into a full architecture.

Three limitations, then: thin validation on real operational data, informal ties to industrial standards, and weak operational embedding. Together they mark out a coherent space to contribute in. AURA-PHM sits in it as an LLM-based diagnostic decision-support system grounded in a stratified knowledge base tied explicitly to international standards, validated on a real wind-farm SCADA dataset with documented failure events, and shaped for operator workflows through traceable reasoning and continuity mechanisms. To set out the

architectural choices behind it, the next subsection looks at what Retrieval-Augmented Generation actually does in technical domains.

2.4 Retrieval-Augmented Generation in Technical Domains

Section 2.3 made a case: LLMs boxed inside structured frameworks, knowledge bases, schemas, validation loops, human oversight, tend to beat unconstrained generators in industrial diagnosis, above all when they run as agents inside structured workflows. Of the patterns that supply that structure, Retrieval-Augmented Generation (RAG) has become the standard way to ground LLM reasoning in verifiable external knowledge. This subsection takes RAG apart: its foundations, how it has evolved, and where it strains in safety-critical industrial use.

RAG was formalised by Lewis et al. [18] in their 2020 NeurIPS paper *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. The original design pairs a pre-trained parametric memory, usually a sequence-to-sequence transformer such as BART, with a non-parametric memory, a dense vector index of a reference corpus reached through a pre-trained neural retriever. The limitations that motivated Lewis et al., framed for open-domain natural language processing (answering general factual questions from corpora such as Wikipedia), read almost as if written for industrial diagnosis: a pre-trained language model cannot easily expand or revise its memory, cannot readily show why it predicted what it did, and may hallucinate [18]. An explicit retrieval step answers all three. Knowledge can be revised and expanded directly, the retrieved context can be inspected and traced, and the generation is tied to documents the user can check independently.

Since 2020, RAG has spread well past its original open-domain question-answering home. For technical and industrial settings, three developments matter most here.

The first is the realisation that industrial knowledge is not uniform, and that one-size retrieval fails technical domains. Fan et al. [14], in their *Synergistic RAG* framework for industrial fault diagnosis, draw a line between two kinds of knowledge that live side by side in maintenance work: *systematic knowledge*, the operation manuals and safety standards, prescriptive, rigid, treated as ground truth; and *experiential knowledge*, the historical maintenance logs and field reports, descriptive, narrative-heavy, treated as heuristic reference [14]. Naive vector RAG, they argue, trips over the exact terminology of systematic knowledge through semantic drift, while pure graph-based RAG is often costly and overkill

for simple factual lookups. Their dual-path design routes fact-seeking queries down a systematic path and diagnosis-seeking ones down an experiential path, and reaches a 91.5% safety pass rate on the proprietary CraneDiagQA benchmark, against 81.2% for standard hybrid RAG on the same set [14]. The lesson recurs throughout the industrial RAG literature: match the retrieval strategy to the epistemic status of the source, and diagnostic quality moves with it.

The second is the marriage of RAG with knowledge graphs (KGs) for multi-hop causal reasoning. As Section 2.3 noted, Nie et al. [16] pair a fault knowledge graph with a fine-tuned GPT-4o through RAG, retrieving entities and reasoning paths to steer fault-mode matching and causal inference while holding hallucination down. Wang et al. [15] generalise the same pattern in *MS-RAG*, a multi-stage retrieval-augmented framework for digital twins that fuses vector retrieval over structured equipment parameters with knowledge-graph traversal over fault-propagation paths, reporting better diagnostic accuracy and interpretability on complex digital-twin scenarios. The point that keeps surfacing: a knowledge graph adds a layer of structural constraint that dense retrieval alone cannot, especially when the reasoning has to walk a chain of cause and effect.

The third is the uncomfortable finding that retrieval can hurt as easily as help when the retrieved context does not match the query. Liu et al. [29], with the *RADAR* framework for aviation fault diagnosis, show it on data: on the C-MAPSS benchmark, naive semantic RAG over an aviation incident knowledge base dragged LLM diagnostic quality down by about 26% against a no-retrieval baseline (an LLM-as-a-Judge score of 6.18 without RAG, 4.58 with naive RAG), because queries about sensor-level anomalies kept pulling back incident reports on unrelated events that merely shared vocabulary [29]. Their answer is a *Physics-Chain-of-Thought* approach that filters retrieved incidents in two stages, subsystem matching, then physics-based consistency scoring, and recovers a 37.6% gain in diagnostic usefulness over the naive baseline once it is in place [29]. The moral is sharp: in safety-critical domains retrieval is not free good, and the filtering, ranking, and constraint machinery around it matters at least as much as the retrieval.

Past these three, a few practical lessons keep recurring across the industrial RAG literature. Schema-constrained outputs, where the LLM fills a structured template instead of writing free-form prose, consistently cut hallucination and make downstream checking easier [13]. Re-ranking the retrieved set beats raw similarity ordering whenever domain relevance and semantic similarity part ways [14]. And validation, whether a pre-retrieval physics filter as

in RADAR [29] or a post-generation self-correction loop with explicit safety constraints as in Synergistic RAG [14], shows up again and again as the last guardrail against an unsafe recommendation.

For all that maturation, RAG in technical domains still runs into open problems, and they frame this thesis's contribution. Most industrial RAG systems treat the knowledge base as one homogeneous corpus, with no explicit sorting of sources by epistemic status, no line drawn between, say, a certified international standard and an operator's field note or a vendor recommendation. The Synergistic RAG framework above [14] is an exception, but its systematic-versus-experiential split is binary; finer stratification, especially anchored to formal international standards, stays rare in published work. Integration with operational workflows, shift continuity, operator feedback, traceable audit trails, is called desirable and then seldom built. And most systems are judged on proprietary benchmarks made by the very authors who designed them, with little cross-comparison and little evidence they generalise to noisy real deployments.

These limits track the gaps from Section 2.3 and drive the architectural choices here. AURA-PHM uses a stratified knowledge base that separates sources by epistemic status across two retrieval pools. The stable technical pool, assembled and curated at design time, holds four strata: a normative-technical stratum anchored to international standards (IEC 61400-4) and to codified failure-mode taxonomies; an empirical stratum built from historical failure events in the operational SCADA dataset; an operational-contextual stratum describing the fleet's normal behaviour and seasonal patterns; and a decision-support stratum encoding differential diagnostic reasoning and operational action protocols. A fifth stratum, the dynamic operational-memory stratum, lives as a physically separate retrieval pool that grows at runtime, gathering the structured shift briefings and operator feedback produced while the system runs (Section 2.7). Its epistemic status is not that of the curated strata, experiential knowledge built through operation rather than reference knowledge fixed at design time, and keeping the pools apart lets each be retrieved, maintained, and updated on its own cadence and its own validation regime. Diagnostic outputs are structured to keep traceability, for each output, back to the specific source documents behind it, which serves both the operator's need to trust it and the downstream evaluation of Section 2.6. Before asking how such systems can be evaluated reliably, though, there is a prior concern that turns acute once LLM outputs start feeding safety-critical decisions: the explainability and trust of the diagnostic reasoning itself. That is where the next subsection goes.

***Table 2.1**. Comparative analysis of representative industrial RAG frameworks for fault diagnosis (2025-2026).*

Framework	Industrial Domain	Retrieval Strategy	Knowledge Structure	Evaluation	Key Result
Nie et al. [16]	CNC machine tools	Fine-tuned GPT-4o + KG-guided RAG	Multimodal KG built from fault cases, sensor data, and maintenance experience	Comparative and ablation experiments in real workshop (6 machines, 8 fault types, 28 samples)	Significant improvement in fault mode matching accuracy and reasoning interpretability vs traditional methods
Synergistic RAG (Fan et al. [14])	Heavy mobile cranes	Dual-path: Hybrid Intent Router routing to BM25/vector for manuals or causal subgraph sampling for logs	Binary stratification: systematic knowledge (manuals, standards) vs experiential knowledge (logs)	CraneDiagQA benchmark (600 queries)	91.5% safety pass rate vs 81.2% standard hybrid RAG baseline; diagnostic utility 4.48/5
MS-RAG (Wang et al. [15])	Industrial digital twins	Multi-stage: vector retrieval over equipment parameters + KG traversal over fault propagation paths	Knowledge graph modelling fault-symptom-cause-maintenance causal chains	Comparative experiments on digital twin scenarios	Improved fault diagnosis accuracy and reduced hallucination rates through causal reasoning
RADAR (Liu et al. [29])	Aviation fault diagnosis	Two-tier architecture: edge TCN autoencoder + offboard cognitive reasoner with Physics-Chain-of-Thought retrieval filtering	ASRS incident knowledge base with subsystem-matched physics consistency scoring	NASA C-MAPSS turbofan + CMU ALFA UAV datasets, LLM-as-a-Judge evaluation	37.6% improvement in diagnostic usefulness via Physics-CoT vs naive RAG, which itself degrades performance by approx. 26%

2.5 LLM Explainability and Trust in Critical Systems

The architectural patterns from Section 2.4, stratified knowledge bases, dual-path retrieval, physics-based filters, self-correction loops, all rest on one quiet assumption: that grounding the generation in retrievable evidence is enough to make an LLM diagnostic system trustworthy. Half right, and incomplete. Retrieval lets the output be traced to source documents, but it says nothing about *how* the LLM weighted, combined, and reasoned over those documents to arrive at the final claim. In a safety-critical system, where an operator has to decide whether to act on a recommendation that might shut a turbine down, the

question is not only “where does this come from?” but “why should I believe it?”. That second question is what the LLM explainability literature takes up.

The fullest recent survey of LLM explainability comes from Luo and Specia [30], who sort the methods into two families by what they set out to explain [30]. *Local analysis* tries to show how the model reached a specific prediction for a given input, and covers feature-attribution methods such as LIME, SHAP, and gradient- and vector-based techniques, along with the dissection of individual transformer components (multi-head self-attention sublayers, MLP sublayers). *Global analysis* goes after the knowledge or linguistic properties baked into the model’s hidden-state activations, through probing methods that train shallow classifiers on internal activations, and mechanistic approaches like circuit discovery, causal tracing, and the vocabulary lens [30]. The survey stresses that explainability does two different jobs: for end users it builds trust by laying out the model’s reasoning in plain terms; for developers and researchers it exposes unintended biases and points to where the model could improve [30]. Two jobs, two stakeholders, and one explanation strategy rarely serves both well.

One point from Luo and Specia bears directly here: sheer scale makes modern LLMs hard to explain. Billions of parameters and vast training data resist interpretation, and traditional methods like SHAP values grow impractical at that size [30]. It lands on this work squarely. AURA-PHM’s L3 reasoning layer runs on GPT-4o-mini, whose internal weights are out of reach and whose attention activations no API user can inspect. Mechanistic interpretability, however rich, is effectively off the table for closed-source commercial LLMs in industrial use. So the practical question moves: not “how do we explain the internal computation?” but “how do we make the diagnostic output transparent enough that a domain expert can check it against external evidence?”.

That move, from internal interpretability to external verifiability, runs through the industrial AI trustworthiness literature too. Ahangar et al. [31], reviewing AI trustworthiness in manufacturing, name four dimensions of trustworthy industrial AI: transparency, fairness, robustness, and accountability [31]. Today’s explainability methods, they note, mostly serve developers and designers, helping with debugging and oversight, while non-expert users such as plant operators or maintenance technicians need something better suited to them [31]. The review walks through widely used toolkits, AIX360, LIME, SHAP, XAITK, Quantus, and weighs their strengths and limits in deployment. One finding keeps returning:

no single toolkit covers all four dimensions, and trustworthy industrial AI ends up needing a mix of techniques at different points in the pipeline.

Within LLM-based fault diagnosis the explainability problem gets more concrete. Qaid et al. [11] argue that an explainable diagnosis asks the model not just to classify correctly but to lay out a reasoning chain a domain expert can audit. Their FD-LLM framework brings in context-aware spectrum language modelling built for exactly that, reasoning-driven fault analysis you can follow, and scores the plausibility of the generated reasoning with LLM-as-a-Judge alongside RAGAS faithfulness metrics [11]. The pairing says something: for industrial diagnostic LLMs, trust hinges not on internal model interpretability but on whether the externalised reasoning can be audited. Tsallis et al. [12] reach the same place, observing across their studies that explainability is treated less as an intrinsic property of the model and more as a property of the framework that constrains, structures, and validates what the model produces.

Past surveys and toolkit comparisons, three practical lessons keep coming up in industrial settings. First, *structured outputs*, JSON schemas, slot-filled templates, formal reasoning chains, make an LLM diagnostic claim far easier to audit than free-form prose [31]. A structured output makes each field separately addressable: the operator can read the failure-mode field, the confidence field, and the cited-source field on their own, instead of picking apart a paragraph. Second, *source attribution* at the level of the diagnostic output, tying the conclusion to the specific document or section in the knowledge base the supporting evidence came from, is essential for downstream verification, all the more when the knowledge base is stratified by epistemic status as in Section 2.4. Third, *deterministic post-processing* of LLM outputs, deriving the operational urgency from a data-driven health index computed independently of the LLM through a fixed rule-based mapping, can act as a transparency layer that caps the operational consequences of any LLM assessment that might otherwise collapse onto a single category.

For all these settled principles, the explainability literature for critical industrial systems keeps a recurring blind spot, and it frames this thesis's contribution. Most of the work targets either the technical interpretability of the model, largely intractable for closed-source LLMs as noted, or post-hoc explanations bolted onto the final output. The middle layer, the traceability of the reasoning chain from retrieved evidence to final claim, with explicit note of which source informed the conclusion and at what epistemic status, gets called desirable and then left underdeveloped. AURA-PHM works that gap by combining structured outputs

(each diagnostic field separately auditable), source attribution at the output level (each output linked to the source documents and stratum behind it), and deterministic post-processing of the data-driven health index, computed independently of the LLM by the Normal Behaviour Model, into operational urgency levels through a fixed rule-based mapping. The aim is not to make the underlying LLM transparent in any mechanistic sense, unreachable with today's commercial APIs, but to make the output traceable enough that a domain expert can check it against the evidence inside the normal workflow. Even so, the match between a cited source and the substance of the generated reasoning stays an emergent property of LLM generation rather than something the architecture can guarantee; that residual *traceability-versus-fidelity gap* is taken up in the Discussion chapter of this thesis.

Trust in these systems, in the end, rests not only on how transparent each individual output is but on how rigorously the system is evaluated as a whole. The next subsection turns to the RAGAS evaluation framework, and to its known weaknesses in settings where the LLM under evaluation and the LLM acting as judge can share systematic biases.

2.6 Evaluating LLM-Based Diagnostic Systems: RAGAS and the Self-Preference Bias

The mechanisms of Section 2.5, structured outputs, source attribution, deterministic post-processing, make individual outputs more auditable, but they give no aggregate read on how the system performs. For an LLM-based diagnostic system to belong in a critical industrial setting, its overall quality has to be measurable: how often does the reasoning stay faithful to the retrieved evidence, how well does the answer meet the diagnostic question, how focused is the retrieved context? Without large-scale human-annotated reference answers, and almost no operational industrial deployment has them, those measurements fall to automated evaluation frameworks. The dominant one is RAGAS.

RAGAS (Retrieval Augmented Generation Assessment), introduced by Es et al. [32], is a framework for *reference-free* evaluation of RAG pipelines. It answers a particular difficulty in judging RAG systems: quality has several distinct dimensions, whether retrieval finds relevant context, whether the LLM uses that context faithfully, and whether the generated answer is any good, and ground-truth human annotations are usually missing [32]. RAGAS offers three core metrics, each computed automatically by prompting an LLM:

Faithfulness gauges whether the claims in the answer can be inferred from the retrieved context. It pulls out the individual statements in the answer, checks each against the context, and scores the fraction that hold up [32].

Answer Relevance gauges whether the answer actually addresses the question. The LLM is prompted to generate candidate questions from the answer, and the cosine similarity between those and the original query is averaged [32].

Context Relevance gauges whether the retrieved context carries only what the question needs, picking out the sentences essential to answering it [32].

The authors check RAGAS against human annotations on the WikiEval dataset, reporting agreement of 95% on faithfulness, 78% on answer relevance, and 70% on context relevance [32]. They say plainly that *"we found context relevance to be the hardest quality dimension to evaluate"*, especially on longer context passages, where the LLM-as-judge struggles to tell which sentences actually matter [32]. Operational RAG evaluations report the same pattern widely.

Since it appeared, RAGAS has been taken up across industrial RAG applications, the LLM-based fault-diagnosis frameworks of Section 2.3 included [11]. Its appeal comes down to three things: it needs no human-annotated reference answers, it drops into an existing RAG pipeline with modest engineering, and it yields interpretable per-metric scores you can track over time as the knowledge base or the LLM changes. For a thesis working with a stratified knowledge base and a single operational dataset, that makes RAGAS the natural primary evaluation framework.

The same properties that make RAGAS practical, though, expose it to a methodological worry only recently pinned down in the literature: the self-preference bias of LLM-as-a-Judge.

Wataoka et al. [33] put forward a quantitative metric for self-preference bias in LLM evaluators and run it over eight different LLMs. GPT-4 comes out with a self-preference bias score of 0.749, the highest of the eight [33]. Chasing the cause, they test whether LLMs favour outputs more familiar to them, as signalled by lower perplexity. The analysis confirms it: *"LLMs assign significantly higher evaluations to texts with lower perplexity than human evaluators, regardless of whether the texts were self-generated"* [33]. The mechanism matters. The bias is not the model recognising its own writing; it is the structural fact that a

model's own outputs tend to carry lower perplexity for that model. It follows from the model's internal probability distribution, not from any deliberate self-flattery.

For RAGAS the consequence is direct. When LLMs from the same family both generate the RAG output and judge its faithfulness, answer relevance, and context relevance, the evaluation is structurally open to self-preference bias. And that is exactly this work's situation: AURA-PHM's L3 reasoning layer generates with GPT-4o-mini, and the RAGAS metrics are computed with the same GPT-4o-mini as judge, the configuration most exposed to the bias, since generator and evaluator are not just related but identical. The 0.749 figure from Wataoka et al. is for GPT-4 rather than GPT-4o-mini, true, but the mechanism underneath, higher scores for outputs of lower perplexity to the model itself, is structural and holds across the GPT family. The point is not that the evaluation must be wrong. It is that the absolute scores have to be read with this systematic effect kept firmly in mind.

Beyond self-preference bias, the literature records several more limits of RAGAS-based evaluation. It is sensitive to how the underlying prompts are written, and small changes in wording can shift the scores [32]. Context relevance, as Es et al. concede, is the hardest dimension to judge, and the LLM-as-judge often falters on longer passages [32]. And the framework measures the dimensions it measures, but on its own it does not capture *operational* dimensions such as how actionable the recommendation is, whether the suggested urgency fits, or how well the diagnostic claim lines up with the failure-mode taxonomy of the underlying industrial standards.

These limits shape the evaluation strategy here. AURA-PHM uses RAGAS as its primary automated framework, computing faithfulness, answer relevance, and context relevance over a curated set of diagnostic queries drawn from the EDP wind turbine SCADA dataset. How the scores are read, though, rests on three explicit commitments. First, the absolute scores are treated as comparative indicators across system configurations, not as absolute measures of quality, given the self-preference bias documented by Wataoka et al. [33]. Second, the weaker showing on context relevance, in line with the difficulty Es et al. acknowledge [32], is read as an evaluator-side limitation rather than a system-side failure, and is examined in the Discussion chapter. Third, the RAGAS metrics are backed by an operational criterion: how well the generated diagnostic claim aligns with the failure-mode classification of IEC 61400-4 [26], a normative anchor that owes nothing to the LLM-as-judge.

That strategy covers the technical performance of the system. A separate concern is whether it works within the operational setting it lands in, whether, specifically, the diagnostic output stays meaningful and actionable across the shift changes and operator handovers that define wind farm maintenance. That operational dimension is the next subsection's subject.

2.7 Operational Continuity and Shift Handover in Industrial Maintenance

Sections 2.4 through 2.6 dealt with how an LLM-based diagnostic system can be grounded in verifiable knowledge, made transparent to the operator, and measured for technical performance. One dimension is still missing. A system that produces a faithful, well-scored output at a single moment can still fail to deliver anything if that diagnostic information is lost, distorted, or pushed down the priority list when responsibility for the asset passes from one operator to the next. In wind farm operations, where remote monitoring centres run around the clock on shift rotations and where a developing anomaly often plays out over days or weeks, keeping diagnostic information continuous across shifts is itself part of the system's safety value. This subsection looks at what is known about shift handover in safety-critical settings, mostly in process industries next door to wind energy, and asks how that knowledge should shape the operational design of the system built here.

The most authoritative review of shift handover in safety-critical industries is Wilkinson and Lardner [34], written in the wake of the 2005 Buncefield oil storage depot explosion and fire in the United Kingdom. They trace the recognition of handover as a critical safety issue back to the 1988 Piper Alpha disaster and its Cullen Inquiry, and document a pattern that repeats across major accidents, Buncefield, Piper Alpha, the BP Texas City refinery accident of 2005, in which breakdowns in shift communication did real work in turning process upsets into catastrophes [34]. Their lesson is not that good handover practice is unknown. It is that well-established guidance is *"still not being implemented fully effectively"* across many high-hazard sectors [34]. The gap is one of application, not of underlying research.

Building on Lardner's foundational work and on Health and Safety Executive (HSE) guidance, Wilkinson and Lardner treat shift handover as one of three elements inside a wider changeover process: *"1) a period of preparation by outgoing personnel, 2) shift handover, where outgoing and incoming personnel communicate to exchange task-relevant information and 3) cross-checking of information by incoming personnel as they assume responsibility*

for the task" [34]. The aim of the handover, they note from HSE guidance, is the accurate and reliable communication of task-relevant information across the change, so that safe and effective working carries on unbroken. The structure matters. Handover is not a single conversational moment but a process that starts with the outgoing operator's preparation and finishes only once the incoming operator has cross-checked what they received and taken on responsibility.

Inside that frame, Wilkinson and Lardner single out several kinds of *high-risk handovers* that deserve extra care: those after a long absence from work, such as crew changes or extended rest; those between experienced and inexperienced staff; those during a plant or process upset; those where maintenance activity straddles the shift transition; and those where safety systems are overridden or otherwise unavailable [34]. These map straight onto wind farm monitoring, where a developing anomaly, unfolding over several shifts by its nature, routinely leaves a process upset spanning a handover, and where the outgoing operator may have spent hours reading a residual pattern the incoming one has never laid eyes on.

The review has more to say about what makes handover arrangements work. Citing Lardner's earlier study of an offshore company's practice, the authors tie successful handovers to the design of handover logs and records, the use of checklists and other prompts, redundancy and diversity in communication media including two-way confirmation, the allowance of adequate time, and an analysis of information needs across operating modes, normal, abnormal, start-up, shutdown, maintenance [34]. The theme underneath: information carried through a single medium, whether purely spoken or purely written, breaks more easily than information sent through several complementary channels. Talk allows questions and clarification; written records persist over longer stretches; structured templates keep key items from dropping out under time pressure.

They also lift an observation from the aviation human-factors literature, via Hobbs, that transfer between shifts works best when it captures *"problems, possible solutions and intentions, and does not just describe what has been accomplished"* [34]. The distinction earns its place. A handover that lists completed actions but drops the interpretive frame those actions sat in passes data, not understanding. The operator who inherits it starts the shift in nominal but not real continuity, and may have to rebuild that frame from scratch as new events arrive.

That interpretive content, the problems, hypotheses, and intentions a handover ought to carry, is mostly tacit: picked up through direct experience, rarely written down. Cárcel-Carrasco and Cárcel-Carrasco, surveying maintenance technicians across several industrial organisations, found tacit operational knowledge running to roughly twice the explicit knowledge on formal record, the gap widest exactly in operational and energy-efficiency actions [21]. Turning that tacit knowledge into explicit, retrievable form has been called one of the most promising uses of foundation models, and yet it stays underexplored in the open literature on LLM-assisted maintenance [22]. It is precisely that conversion the operational-memory mechanism of this thesis sets out to support.

Bringing these observations to LLM-based diagnostic systems in wind turbines means owning a methodological transposition. Wilkinson and Lardner, and the literature behind them, come out of process industries, oil and gas, nuclear, aviation, patient safety, not wind energy as such. Carrying the principles over to remote wind farm monitoring rests on a structural analogy rather than a direct line of evidence: in both, safety-critical decisions are taken by operators on shift rotations, with information about evolving conditions that has to survive the changeover. Read that way, two points stand out. First, the outputs of such a system are exactly the kind of interpretive information, problems identified, hypotheses weighed, actions recommended, that Wilkinson and Lardner mark as critical to handover and most easily lost when the changeover is poorly structured. A system that surfaces a developing anomaly on a night shift is worth something only if that information reaches the morning operator in interpretive form. Second, the structured handover requirements the review documents, preparation, exchange, cross-checking, redundancy across media, capture of intent and not just of action, hand us a concrete operational specification for what an LLM-based diagnostic system ought to support, past the per-query reasoning the earlier subsections dwelt on.

For all its maturity in high-hazard process industries, the handover literature applies only so far to wind farm monitoring, for want of anything written specifically about it. Handover practice in remote monitoring centres varies from operator to operator, tends not to be formalised in published frameworks, and enjoys no codified industry standard of the kind that governs gearbox design (Section 2.2). Folding automated diagnostic outputs into structured handover, marrying the interpretive content an LLM system produces to the procedural rigour of the human-factors literature, is thus an operational design space that agent-oriented LLM work has acknowledged [10], [12] but rarely worked out in any detail.

That gap drives a specific design choice here. AURA-PHM includes an automated shift-continuity mechanism that produces, at fixed intervals matching the typical 8-hour shift rotation of remote wind farm monitoring centres, a structured briefing: the diagnostic events seen during the shift, the open items that need continued monitoring by the incoming shift, and the residual patterns whose evolution should be watched over the coming hours. The briefing is generated as a structured document and kept in the dynamic operational-memory stratum of the knowledge base, the runtime retrieval pool, architecturally distinct from the four curated strata of Section 2.4, that gathers shift briefings and operator feedback, so the information stays retrievable for the next shift's diagnostic queries. The intent is to give the human handover conversation a reliable substrate to run on. The system automates the *preparation* and *exchange* elements of the three-part changeover by assembling the relevant diagnostic information, while *cross-checking* stays with the human operator taking the next shift. As it stands, the system does not capture operator-side actions and their outcomes, nor does it offer a fully conversational handover interface; both are natural extensions, discussed in the conclusions chapter.

Across Sections 2.1 through 2.7, then, the literature establishes both the scientific maturity of wind turbine condition monitoring and the architectural, evaluative, and operational space in which a new generation of LLM-based diagnostic systems is taking shape. The next subsection gathers the gaps these reviews leave and sets the thesis's contribution against them.

2.8 Research Gaps and Positioning of the Present Thesis

The reviews in the preceding subsections trace one coherent line across complementary domains. SCADA-based anomaly detection in wind turbines has reached methodological maturity but stops before structured diagnostic interpretation; international standards codify failure-mode taxonomies and condition-monitoring frameworks but prescribe no operational logic for the diagnostic decision; LLM-based industrial diagnostic systems have appeared fast across several research lines but run into limits in validation, integration with codified standards, and operational embedding; Retrieval-Augmented Generation has become the standard way to ground LLM reasoning in verifiable knowledge, yet is usually applied to homogeneous corpora with no explicit stratification; explainability research has moved from internal interpretability, largely out of reach for closed-source commercial models, toward

external verifiability of the output; automated frameworks like RAGAS give reference-free quality assessment but are structurally open to self-preference bias when generator and evaluator share a model; and the human-factors literature on shift handover, developed for process industries next to wind energy, sets out operational principles rarely folded into automated diagnostic frameworks.

Put together, the reviews mark out a coherent space of methodological gaps that this thesis takes on. They are not isolated limitations of individual systems but interconnected features of how LLM-based industrial diagnostic systems get designed, evaluated, and deployed today. Five gaps matter most here. They sharpen the three macro-gaps from Section 1.3: Gap 1 and Gap 2 answer directly to the first two, Gap 5 to the third, while Gap 3 and Gap 4 come out specifically of the review of how existing systems are validated and evaluated.

Gap 1 Stratified knowledge bases anchored to international standards. The industrial RAG literature of Section 2.4 shows that homogeneous knowledge bases fail technical domains where sources of different epistemic status, certified standards, peer-reviewed evidence, operational records, protocol guidelines, sit together and need to be retrieved differently. The Synergistic RAG framework of Fan et al. [14] draws a binary line between systematic and experiential knowledge, but finer stratification, one where international standards such as IEC 61400-4 form a distinct stratum with their own retrieval and citation logic, stays largely undeveloped in published work. This thesis answers the gap with a knowledge base split into five strata of different epistemic status across two retrieval pools. Four curated strata make up a stable technical pool assembled at design time: a normative-technical stratum anchored to IEC 61400-4 and codified failure-mode taxonomies; an empirical stratum built from historical failure events in the EDP wind turbine SCADA dataset; an operational-contextual stratum describing normal turbine behaviour and seasonal patterns; and a decision-support stratum encoding differential diagnostic reasoning and operational action protocols. The fifth is a dynamic operational-memory stratum, a separate retrieval pool that accumulates structured shift briefings and operator feedback at runtime; its role in continuity is developed under Gap 5.

Gap 2 Traceable diagnostic outputs in safety-critical industrial systems. The explainability literature of Section 2.5 keeps hitting one limitation: most of it goes after either the technical interpretability of the model, largely intractable for closed-source LLMs like the GPT-4o-mini used here, or post-hoc explanations pinned to the final output. The layer in between, the traceability of the reasoning chain from retrieved evidence to final claim, stays

underdeveloped. This thesis works that gap with three complementary mechanisms: structured outputs in which each diagnostic field is separately auditable; source attribution at the output level, tying the conclusion to the source documents and stratum behind it; and deterministic post-processing of the data-driven health index, computed independently of the LLM by the Normal Behaviour Model, into operational urgency levels through a fixed rule-based mapping. That last mechanism also heads off a structural risk peculiar to LLM classification under constrained label sets: an LLM, unsure of a multi-class call in a safety-critical context, tends to fall back on the most cautious category, which flattens the discriminative value of the urgency assessment. Deriving urgency from a data-driven quantity computed outside the LLM sidesteps that. The residual traceability-versus-fidelity gap, where the match between cited source and the substance of the reasoning stays an emergent property of generation rather than a guarantee of the architecture, is taken up in the Discussion chapter.

Gap 3 Validation on real operational SCADA data with documented failures. Most LLM-based diagnostic systems in Section 2.3 are judged on idealised or proprietary benchmarks, with little validation on production-scale SCADA from working wind farms. The Tan and Carroll study [3] is one of the few working directly with labelled turbine SCADA, and even it stays at binary classification rather than multi-failure-mode reasoning. This thesis answers the gap by validating on the EDP wind farm SCADA dataset, a publicly available operational record of four turbines monitored across 2016 and 2017, and by using a documented failure event (T06 gearbox bearings, 17 October 2017) as the post-hoc reference for the diagnostic chain. During detection the system runs blind to the failure log; the log is used only afterwards, to assess whether the system surfaced the developing anomaly in the run-up to the certified failure.

Gap 4 Evaluation strategies that acknowledge LLM-as-judge bias. RAGAS, reviewed in Section 2.6, has become the standard way to assess RAG-based diagnostic systems, but its usual application is structurally open to the self-preference bias Wataoka et al. document [33], the more so when generator and evaluator are the same model, as they are here, with GPT-4o-mini serving both for L3 reasoning and as the LLM-as-judge for RAGAS. This thesis meets that exposure with three explicit commitments: absolute scores read as comparative indicators across configurations, not as absolute measures of quality; a weaker showing on context relevance read as an evaluator-side limitation, in line with the difficulty Es et al. acknowledge [32]; and the RAGAS metrics backed by an operational criterion, the

alignment between the generated diagnostic claim and the failure-mode classification of IEC 61400-4 [26], a normative anchor independent of the LLM-as-judge.

Gap 5 Operational embedding through structured shift continuity. The literature of Section 2.7 establishes shift handover as a critical safety dimension in process industries next to wind energy, yet folding automated diagnostic outputs into structured continuity has rarely been worked out in published frameworks. This thesis answers the gap with an automated shift-continuity mechanism, set out in detail in the methodology chapter, that generates, at fixed intervals matching the typical 8-hour shift rotation of remote wind farm monitoring centres, a structured briefing of diagnostic events, open items, and residual patterns to track. The briefing lives in the dynamic operational-memory stratum introduced under Gap 1, supporting the preparation and exchange elements of the three-part changeover documented by Wilkinson and Lardner [34].

These five gaps are not independent. The stratified knowledge base (Gap 1) is the architectural substrate that lets source attribution be traceable (Gap 2); the choice of a real operational dataset with documented failures (Gap 3) gives the empirical anchor against which traceability and evaluation (Gap 2, Gap 4) can be judged; the operational embedding through shift continuity (Gap 5) carries the traceability of single outputs up into a continuity property of the whole system. The contribution of this thesis is not any one technical innovation but the *integration* of these five elements into one coherent diagnostic decision-support framework, AURA-PHM, validated on a real wind farm SCADA dataset and grounded explicitly in international standards.

That closes the literature review. The next chapter sets out the methodology and architecture of the AURA-PHM framework in detail, building on the foundations laid here.

***Table 2.2**. Summary of the five research gaps identified in the literature reviewed across Sections 2.1-2.7 and the corresponding contribution of the present thesis.*

Gap	Limitation in the Literature	Ref.	Contribution of the Present Thesis
Gap 1 **Stratified KB anchored to standards**	Industrial RAG systems typically treat knowledge as a homogeneous corpus; stratification, where attempted, is binary (systematic vs experiential). Fine-grained stratification anchored to	§2.4	Five-stratum knowledge base in two retrieval pools: a curated technical pool with four strata, normative-technical (IEC 61400-4, ISO 20816-21), empirical (historical EDP failures), operational-contextual (normal turbine behaviour), decision-support (diagnostic and action protocols), plus a dynamic operational-

Gap	Limitation in the Literature	Ref.	Contribution of the Present Thesis
	international standards is largely undeveloped		memory stratum (shift briefings and operator feedback) as a separate runtime pool
Gap 2 **Traceable diagnostic outputs in safety-critical systems**	Explainability work focuses either on internal model interpretability (intractable for closed-source LLMs) or on post-hoc output explanations; the intermediate layer of reasoning-chain traceability remains underdeveloped	§2.5	Three complementary mechanisms: structured outputs (per-field auditability), output-level source attribution, deterministic post-processing of HI computed independently by the NBM into operational urgency levels
Gap 3 **Validation on real SCADA with documented failures**	LLM-based diagnostic systems are predominantly evaluated on idealised or proprietary benchmarks, with limited validation on production-scale wind farm SCADA data	§2.3	Validation on the EDP wind farm SCADA dataset (4 turbines, 2016-2017); documented failure event (T06 gearbox bearings, 17 October 2017) as post-hoc validation reference; system blind to failure log during training and detection
Gap 4 **Evaluation strategies acknowledging LLM-as-judge bias**	RAGAS-based evaluation is structurally exposed to self-preference bias, especially when generator and evaluator share the same model, the maximally exposed configuration	§2.6	Three explicit methodological commitments: absolute scores as comparative indicators only; low context relevance interpreted as evaluator-side limitation; RAGAS complemented by normative anchoring to IEC 61400-4
Gap 5 **Operational embedding through shift continuity**	Integration of automated diagnostic outputs into structured shift handover processes is acknowledged as desirable but seldom developed in published frameworks	§2.7	Automated shift continuity mechanism producing structured briefings at intervals matching the 8-hour shift rotation, retained in the dynamic operational-memory stratum of the KB; supports preparation and exchange of the three-element changeover process of Wilkinson & Lardner [34]

3. The AURA-PHM Framework: A Layered Architecture for Traceable Diagnostic Decision Support

This chapter presents AURA-PHM, the architectural contribution of the thesis. It describes the framework in domain-general terms: what its layers and components do, how they cooperate through their interfaces, and the engineering decisions that shaped them.

3.1 Architectural overview and design principles

AURA-PHM is built as three cooperating functional layers, as flagged in Section 1.5. The *ingestion and detection* layer turns raw operational data into a quantitative health assessment through a fully deterministic chain. The *diagnostic reasoning* layer takes that assessment, on alarm, and renders it as a structured, source-grounded diagnosis, using Retrieval-Augmented Generation over a stratified knowledge base. The *human oversight and persistence* layer hands control of the output to the operator, records every decision in auditable form, and turns operator-generated knowledge into retrievable content for later sessions. *Figure 3.1* lays out the full architecture and the information flow that ties the three layers together.

One clarification, before the layers are taken one at a time. They are functional constructs, not separate pieces of software: the prototype realises them as a small set of cooperating Python components run in sequence, and several components serve more than one architectural concern. Even so, the layered view is the right level of abstraction for this chapter, because it captures the division of responsibilities the implementation enforces, what each part is allowed to decide, on which inputs, and with which guarantees. The mapping from functional layers to concrete implementation units is spelled out in Chapter 4.

Four design principles govern the architecture. They recur through the component descriptions of Sections 3.2 to 3.4, and each answers directly to a research gap consolidated in Section 2.8.

The first is *traceability by design*. Traceability is not a reporting feature bolted on after generation; it is a structural property enforced at every stage. Each diagnostic output carries the identifiers of the specific knowledge fragments that supported it, the numerical indicators that triggered it, and the parameters it was produced under, and each is written to a persistent audit trail through a single, controlled update path. The aim, following the requirement under

Gap 2, is that an operator, a maintenance manager, or an external auditor can reconstruct after the fact the complete evidential basis of any single diagnosis.

The second is *epistemic separation*. Knowledge sources of different epistemic status are never poured into one undifferentiated retrieval space. The knowledge base is stratified into five strata across two retrieval pools: a curated technical pool, whose four strata, normative-technical, empirical, operational-contextual, and decision-support, are filled at design time and change only through deliberate manual curation; and a dynamic operational-memory stratum, filled on its own at runtime by shift briefings and operator feedback. Inside the technical pool, the strata are realised as distinct, individually identified sources, so the provenance of every retrieved fragment stays attributable to the stratum it came from; retrieval itself works at the granularity of the two pools. This separation, the answer to Gap 1, keeps provenance intact and lets each pool be maintained on its own update cadence.

The third is a strict *deterministic–probabilistic division* of responsibility. Anything that can be done deterministically is: data conditioning, anomaly detection, the synthesis of the Health Index, alarm triggering, and the mapping of the Health Index onto an operational urgency level are all rule-based and reproducible. The large language model is kept to the functions that are inherently interpretive or synthetic, articulating a structured diagnosis from retrieved technical knowledge, answering operator queries grounded in the knowledge base, drafting shift-handover briefings, and it is never the sole authority on a safety-relevant operational parameter. Where the model does voice such a parameter, urgency being the case in point, its output is kept beside the deterministic value for comparison rather than acted on, a choice driven by a failure mode seen empirically while the system was being built and documented in Chapter 4.

The fourth is *human primacy*. AURA-PHM supports decisions; it does not make them. No output triggers an autonomous action, and every diagnosis goes to the operator for review. The principle reaches past simple human-in-the-loop validation, though. The operator's confirmation, correction, and free-text observations are captured, persisted, and re-indexed into the operational-memory stratum as retrievable fragments, through the same semantic retrieval that serves the technical pool, so that the knowledge produced by human oversight joins the evidential basis available to later diagnostic sessions and later shifts. That closes the loop visible in *Figure 3.1* and instantiates, at the architectural level, the tacit-to-explicit knowledge conversion under Gap 5. The system is built as a loop, not a pipeline, and the difference is structural rather than rhetorical.

The rest of the chapter takes each layer in turn. Section 3.2 covers the ingestion and detection layer and its three components; Section 3.3 the diagnostic reasoning layer, knowledge base and reasoning engine both; Section 3.4 the human oversight and persistence layer. Section 3.5 then follows the full information flow across the layer boundaries, and Section 3.6 draws the architectural choices together and sets up their instantiation.

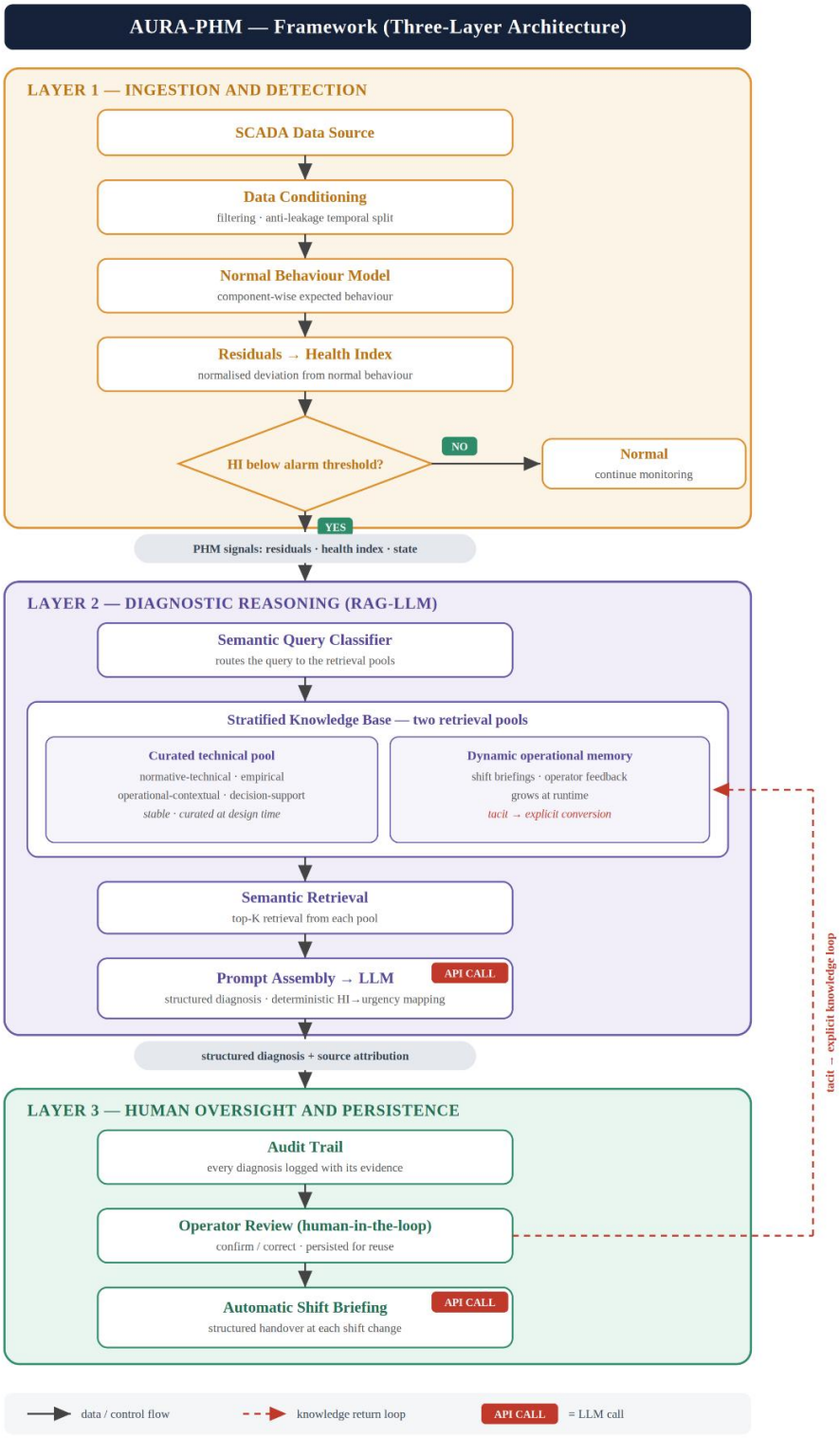

***Figure 3.1**. The AURA-PHM framework.*

3.2 The ingestion and detection layer

The ingestion and detection layer is the deterministic foundation of the framework. Its job is to turn a raw stream of supervisory control and data acquisition (SCADA) records into an

interpretable per-record health assessment, a scalar index backed by the full per-signal evidence behind it, and to decide, by rule rather than by inference, when that assessment deserves diagnostic attention. Nothing probabilistic or generative runs in this layer. Every transformation is reproducible, every parameter is fixed before deployment, and every quantity it produces is kept for downstream traceability. The layer has three components, taken in turn below. One constraint shapes all three: the layer works *blind*, in the sense that nothing about the location or timing of historical failures reaches it at any point, and normality is learned only from data that precede the evaluation stream. Anomalies have to surface from the data alone, a protocol whose validation implications are discussed in Chapter 5.

3.2.1 Data ingestion and conditioning

The first component takes in the raw SCADA archive and cuts it down to the records on which normal-behaviour modelling actually means something. Its input is the unprocessed archive of ten-minute aggregated channels for the monitored asset; its output is a conditioned data frame holding only valid records captured inside the productive operating regime.

Three operations define the component. First, records are filtered to the asset's operational envelope: anything outside the productive wind-speed window, or with active power below a minimal production threshold, is dropped [25]. Second, records missing any value the detection model needs, the operational input features and the monitored target signals, are removed. Third, the component enforces the temporal integrity of the training–test split: the data used to characterise normal behaviour come strictly from a historical period ahead of the evaluation stream, a constraint the implementation enforces structurally rather than by convention.

The main engineering decision here is the restriction to the productive regime. A Normal Behaviour Model characterises how health-relevant signals respond to operating conditions *during normal operation*; records taken during standstill, start-up, or curtailment reflect regimes where those response relationships do not hold, and letting them in would contaminate both training and residual evaluation. The filter is therefore not data cleaning in the generic sense but a modelling precondition. The second decision is to rely solely on standard ten-minute SCADA aggregates, already collected for operational purposes on the assets in question: the framework asks for no extra sensing hardware, in line with the accessibility argument of Section 1.2.

3.2.2 The Normal Behaviour Model

The second component does anomaly detection proper, under the Normal Behaviour Modelling paradigm reviewed in Section 2.1 [6], [23]. Its input is the conditioned data frame; its output, for every complete record and every monitored signal, is the expected value of that signal under normal behaviour and the residual between measurement and expectation. Completeness is guaranteed upstream by the conditioning component, but the model still guards against incomplete records at evaluation time, as a defensive measure.

The component is an ensemble of independent regression models, one per monitored signal. Each learns, from healthy historical data only, the mapping from a shared set of operational variables, describing the environmental and production state of the asset, to the expected value of its target signal. At evaluation time, the residual r(t) = measured(t) − expected(t) measures how far the asset has drifted from its own learned normality. Alongside each trained model, the component keeps a small set of calibration constants computed on the training residuals, among them the maximum absolute training residual of each signal, which becomes the fixed normalisation reference for the health assessment component.

Three engineering decisions define the component. The first is committing to the normal-behaviour paradigm itself, rather than supervised failure classification: as Sections 2.1 and 2.2 established, labelled failure examples are scarce in real fleets, whereas a model of normality needs only healthy operating history, which every running asset accumulates by definition. The second is the choice of *independent per-signal models* over a single multivariate one. The deciding argument is diagnostic, not predictive: per-signal residuals pin the anomaly on specific physical components, and that attribution is exactly the numerical evidence the diagnostic reasoning layer will read. A monolithic anomaly score would detect but could not localise. When the evidence has to name one most-anomalous component, to focus the diagnostic query and to fill the audit record, it is the signal whose residual is largest relative to its own calibrated maximum, the same per-signal reference used below to normalise the Health Index. Attribution rests on each deviation measured against its own learned baseline rather than on absolute magnitude, so a signal with an intrinsically wide normal range cannot dominate on scale alone. The third is the regression family, ensemble decision-tree regressors, chosen for their ability to handle non-linear interactions among operational variables and for a modest tuning burden; the specific configuration and its training procedure are reported in Chapter 4.

3.2.3 Health assessment and alarm generation

The third component fuses the per-signal residuals into a scalar Health Index and runs the deterministic alarm logic [6]. Its input is the set of residuals and calibration constants from the Normal Behaviour Model; its outputs are the Health Index time series, a discrete asset state, and a binary alarm flag, which is the activation interface of the diagnostic reasoning layer.

The Health Index at each evaluated record is given by *Equation (3.1).*

$$HI(t)=1-\sum_{i} w_i \cdot \min\left(\frac{|r_i(t)|}{r_i^{max}}, 1\right) \tag{3.1}$$

Here $r_i(t)$ is the residual of signal i, $\hat{r}_i$ its fixed calibration maximum from the training period, and w_i are component weights summing to one, set to reflect the relative criticality of the monitored components (their values for the validation case are given in Chapter 4). Since each normalised deviation is capped at one and the weights sum to one, the index is bounded in [0, 1] by construction, with 1 meaning perfect agreement with learned normal behaviour. Two fixed thresholds, an attention threshold and, below it, an alarm threshold, cut the range into three states: normal, attention, and alarm. Crossing the alarm threshold raises the binary alarm flag that wakes the diagnostic reasoning layer.

Four engineering decisions are worth noting. First, *scalar synthesis with evidence preservation*: the Health Index hands the operator a single legible indicator, but the per-signal expected values and residuals behind it are appended to the data stream rather than thrown away, so nothing is lost between detection and diagnosis. Second, *normalisation against fixed calibration maxima*: each residual is scaled by the worst deviation seen during healthy operation, which makes signals of different physical magnitude comparable, and the reference is deliberately frozen at calibration time, the yardstick for what counts as ‘anomalous’ cannot drift with the data being judged, a direct application of the reproducibility the traceability principle demands. Third, *clipping of each normalised contribution at one*: a single badly deviating signal can drain at most its own weight from the index, which stops one extreme reading from saturating the assessment and keeps the weighting scheme interpretable. Fourth, *deterministic alarm logic*: the escalation from monitoring to diagnosis is a fixed-threshold rule, not a learned or generative judgement, in strict application of the third design principle. The question “should an expert look at this?”

is settled by arithmetic; only the question “what does this mean?” is handed to the reasoning layer.

The interface this layer offers the next is therefore minimal and wholly numerical: the conditioned record, its expected values and residuals, the Health Index, the asset state, and the alarm flag. What crosses the boundary is evidence, not interpretation.

3.3 The diagnostic reasoning layer

The diagnostic reasoning layer wakes when the detection layer raises an alarm, or when an operator puts a query to the system. Its job is to turn numerical evidence into a structured, source-grounded diagnostic statement, and to do so under the constraints the design principles set: every retrieved piece of knowledge stays attributable to its source and stratum, and the layer is deterministic wherever determinism is possible and openly declared probabilistic where it is not. It has two components: the stratified knowledge base, which organises what the system knows, and the diagnostic reasoning engine, which governs how that knowledge is retrieved and made into a diagnosis.

3.3.1 The stratified knowledge base

The knowledge base holds the diagnostic knowledge available to the framework, organised, per the second design principle, into five strata of different epistemic status across two retrieval pools. At construction time its input is a set of plain-text knowledge sources; at query time its output is a set of text fragments, each carrying its source identity, its position within the source, and its pool membership.

The curated technical pool holds the four design-time strata: the normative-technical stratum, grounded in certified engineering standards; the empirical stratum, drawn from documented failure history and the residual signatures tied to it; the operational-contextual stratum, describing the normal operating envelope of the monitored assets; and the decision-support stratum, encoding the mapping from diagnostic findings to recommended actions. These strata are realised as distinct, individually identified sources: the stratification is a curated organisation of knowledge by source identity, not a further structural partition of the storage, the boundary enforced by construction is the split between the two pools. The dynamic operational-memory stratum makes up the second pool. It is empty at deployment and fills on its own at runtime from the artefacts of the human oversight layer, shift briefings and operator feedback. Those artefacts are persisted as files and re-indexed into the pool,

their semantic representations computed in the same flow, so they become retrievable within the running session the moment they are written; across sessions, persistence rests on the files themselves, re-ingested at start-up.

Mechanically, every source gets the same treatment at ingestion. Documents are cut into overlapping fragments, so local context survives the segment boundaries; each fragment records its source and sequential position, and pool membership follows from the declared identity of the source. Each fragment is then projected into a semantic embedding space, and the representations are cached, so the cost of representation is paid once per knowledge-base revision rather than once per query. At every load, a coherence check confirms that each fragment collection lines up exactly with its embedding matrix, and flags any misalignment before the system will take queries.

Three engineering decisions define the component. The first is the *separation of pools at the storage level*, rather than as a filter applied at query time: the two pools are physically distinct collections with distinct embedding matrices, so the epistemic boundary of Gap 1 is enforced by construction and cannot be quietly bypassed by a retrieval bug or a ranking artefact. The second is *provenance-preserving fragmentation*: since attribution in AURA-PHM works at the level of each diagnostic output (Gap 2), the unit of retrieval has to be attributable in itself, every fragment that reaches the generative model carries the identity the audit trail needs, and overlapping segmentation makes sure that attributability is not bought at the cost of broken local context. The third is the *single shared embedding space*: queries, technical fragments, and operational-memory fragments are all represented by the same embedding model, which makes similarities comparable across pools and is exactly what lets runtime-generated knowledge be retrieved by the same mechanism as curated knowledge, the precondition for the loop the human oversight layer closes.

3.3.2 The diagnostic reasoning engine

The reasoning engine governs the path from activation to structured output. Its input is either the numerical evidence package from the detection layer, residuals, Health Index, asset state, or a free-text operator query; its output is a structured diagnostic record that carries, alongside its content, the full set of retrieved fragments with their identities and scores, plus the retrieval metadata. Attribution, in keeping with how Gap 2 is framed, works at the level of the diagnostic output as a whole: the record documents which evidence informed the diagnosis, not a field-by-field mapping of individual claims to individual fragments.

The engine runs in four stages. The first is *query understanding*: the incoming query is embedded and compared, by cosine similarity, against pre-computed centroids for three query categories, technical, operational, and diagnostic, each centroid built from a set of prototype examples. The mechanism is deliberately lightweight, an example-based engineered heuristic rather than a statistically trained classifier. Its dependability therefore rests on two explicit guards: a minimum absolute confidence on the best-matching category, and a minimum margin between the first and second candidate. If either check fails, or if embedding the query fails, the engine falls back to the diagnostic category, the broadest retrieval configuration, so classification uncertainty pushes the system toward more evidence, never toward less.

The second stage is *pool routing and retrieval*. Technical and operational queries are served from their own pools; diagnostic queries draw on both, taking a fixed quota of technical fragments and a smaller, threshold-gated quota of operational-memory fragments, so runtime memory can inform a diagnosis without crowding out curated technical knowledge, and weakly relevant memory is shut out entirely by an admission threshold on its retrieval score. Retrieval is semantic by default, cosine similarity over the cached embeddings, while a lexical mode and a hybrid mode, which blends the two rankings with fixed weights after normalisation, are supported as alternative configurations; their role as baselines in the evaluation is discussed in Chapter 5. A redundancy-controlled selection finishes the stage: a candidate fragment is dropped if it coincides with, or sits immediately next to, a fragment already selected from the same source, which stops the overlap introduced at segmentation from packing the context with near-duplicates and forces the retrieved evidence to stay diverse.

The third stage is *constrained generation*. The engine assembles a prompt holding the numerical evidence package and the retrieved fragments, each shown with its source identity, and submits it to the large language model in constrained JSON mode at near-zero temperature. The structured response gives the probable diagnosis, the root cause, the recommended action, the model's own urgency assessment, the principal source relied on, and a declared limitation, a field in which the model has to state the boundaries of its own assessment. Two properties deserve emphasis. Constrained generation guarantees the structural validity of the output by construction, which does away with fragile post-hoc parsing; but it has to be said just as plainly that structural validity guarantees nothing about the quality of the reasoning inside the structure, which turns on the quality of the retrieved

context, a dependency that drives the retrieval-quality focus of the evaluation in Chapter 5. In the same way, the near-zero temperature setting minimises stochastic variation across calls but does not make the generative step deterministic, and the framework makes no such claim: reproducibility in this layer is pursued through recording, not through an assumption of determinism.

The fourth stage is *deterministic post-processing* and recording. Independently of the model's urgency assessment, the engine derives the operational urgency from the Health Index through the fixed rule the third design principle anticipated, and both values, the deterministic one, which is authoritative, and the model's, which is observational, are kept side by side in the output. The retrieval query and the retrieval configuration used are recorded in the same structure, so that before the record leaves the layer it already carries what the audit function of the human oversight layer will need: a diagnosis exits the reasoning engine already holding everything required to reconstruct how it was produced.

3.4 The human oversight and persistence layer

The human oversight and persistence layer is where the first and fourth design principles, traceability and human primacy, turn into working mechanisms. It receives the structured diagnostic record from the reasoning engine and carries three responsibilities: recording every diagnostic event so it can be reconstructed in full, putting every diagnosis to operator review, and moving the knowledge produced by detection, diagnosis, and review across the boundary that most reliably scatters it, the shift change. The layer has two components.

3.4.1 Traceability and audit

The audit component persists every diagnostic event in the system. Its input is the structured diagnostic record, together with the numerical evidence that triggered it; its output is the audit trail, a persistent tabular archive in an open, directly inspectable format.

Each audit record brings together, in a single entry, the four families of information needed to reconstruct a diagnostic event after the fact: the identification of the event, system time, asset, and the timestamp of the originating measurement; the numerical evidence, Health Index, asset state, the most anomalous monitored component and its residual; the diagnostic content, probable diagnosis, recommended action, principal source, and both urgency values, the deterministic and the model-generated one, side by side; and the production metadata, the retrieval configuration used, the identities and scores of every knowledge fragment

supplied to the generative model, and the resources consumed. Two review fields round out the record, initialised as pending.

The archive's lifecycle is deliberately plain, and worth stating exactly: records are appended and never removed, and the only mutation the design allows is resolving the review fields of a pending event, which move, on operator review, from pending to the reviewed outcome and the operator's note. So the archive is not immutable in the strict sense of an append-only ledger. It is append-structured, with a single, controlled, semantically meaningful update path through which no past diagnostic content is ever altered. That is a property of the designed update discipline, not an enforced immutability guarantee of the underlying storage, and it is presented as such.

Two engineering decisions define the component. The first is the *co-location of detection evidence and retrieval evidence in the same record*: because each entry carries both the numerical indicators that triggered the diagnosis and the identities of the fragments that informed it, the archive supports not just the reconstruction of individual events but the correlation, across events, between diagnostic quality and the retrieval that produced it, a property the evaluation of Chapter 5 uses directly. The second is the choice of the audit trail as the *single source of record* for everything downstream: operator review writes its outcome into it, and shift briefings are generated from it, so there is exactly one place where the system's history lives, and every derived artefact can be traced back to it.

3.4.2 Operator interaction and shift continuity

The second component closes the loop announced in Section 3.1. Its inputs are the pending diagnostic events and the human operator's judgement; its outputs are resolved audit records, feedback artefacts indexed into the operational memory, and shift-handover briefings.

Operator review pairs a structured outcome with free expression. For each diagnosis, the operator either confirms the system's assessment, corrects it and states the actual cause, or defers, leaving the event pending in the archive; a free-text note can ride along with either substantive outcome. The review then runs through three channels. First, it resolves the review fields of the matching audit record, so the archive permanently tells validated diagnoses apart from corrected ones. Second, it produces a feedback artefact, a dated document holding the outcome, the correction where there is one, and the operator's note, persisted as a file and re-indexed into the operational-memory pool, its semantic representation computed in the same flow, which makes the operator's judgement retrievable

by the diagnostic engine from that point on. Third, since the next briefing is generated from the audit window in which the review happened, the operator's judgement needs nothing more to reach the incoming shift: it travels through the audit record itself, and the briefing, by design, reproduces operator notes verbatim in a dedicated section.

Shift continuity is the component's second function. At fixed shift boundaries, it assembles a structured briefing from the audit trail: the anomalies seen during the closing shift, the diagnoses issued, the review outcomes, and the anomalies that remain open. That open set is not limited to the closing shift. It is computed cumulatively over the whole audit history, so an anomaly first raised several shifts earlier keeps appearing in every later briefing until an operator review closes it with a correction, and reappears if a previously closed component alarms again. Because developing faults, by their nature, play out over multiple shifts, the very condition named in Section 2.7, this carry-forward is what stops a slowly maturing anomaly from dropping silently out of the handover once the shift that first saw it has ended; for the same reason, the open set is surfaced even in shifts that register no new alarm of their own. A division of labour inside the briefing reflects the third design principle: the open-anomaly set is derived deterministically from the audit records, by a fixed rule applied to each component's review history, while the large language model is kept to the narrative synthesis of the shift's events. What remains open is thus answered by arithmetic over the record, not by the model's recollection, so a generative omission cannot drop an unresolved anomaly from the handover. The briefing is persisted three times over: as a human-readable handover document, as an entry in a persistent briefing history, and as indexed fragments in the operational-memory pool, so the incoming operator can read it and the diagnostic engine can retrieve it from the first query of the new shift. It is worth saying plainly what grounds the whole mechanism: the briefing is a synthesis of the audit trail, not of the model's own memory, and the recorded events stay available for verification underneath the summary.

Three engineering decisions complete the picture. The first is the *pairing of structured outcome and free text* in the review: the binary outcome is machine-usable and statistically aggregable, while the free-text note catches exactly the nuance the structured fields lose, the two faces of the tacit-to-explicit conversion Gap 5 demands. The second is the *routing of all human-generated knowledge into the gated operational pool*: feedback and briefings enter the same embedding space as curated knowledge and are retrieved by the same mechanism, but only through the threshold-gated, bounded quota of Section 3.3, human knowledge

informs future diagnoses without ever displacing the certified technical strata. The third is the *human gate on knowledge growth*: the operational memory grows only through review and briefing, that is, only through acts of human judgement. A deployment where operators never review would leave the memory empty and the system functionally identical to its day-one configuration; the framework's capacity to improve is, by construction, a function of the oversight it gets. This is human primacy at its strongest: the human is not a checkpoint bolted onto the pipeline but the mechanism through which the system accumulates any knowledge at all.

With this layer, the loop of *Figure 3.1* closes: a deviation that entered the framework as a numerical residual leaves it as recorded, reviewed, retrievable organisational knowledge.

3.5 Information flow and inter-layer interfaces

The preceding sections described the seven components one by one; this one looks at what crosses the boundaries between them, because the discipline of those crossings is where the design principles of Section 3.1 are either kept or lost. *Figure 3.2* traces the full path of a single anomalous sample through the framework, from its arrival in the data stream to the recorded, reviewed diagnosis. The figure shows the nominal flow, a conceptual schema of the designed path rather than a literal trace of execution. The protective branches described earlier, classification fallback, admission gating, deferred review, error handling, stay active around every step shown.

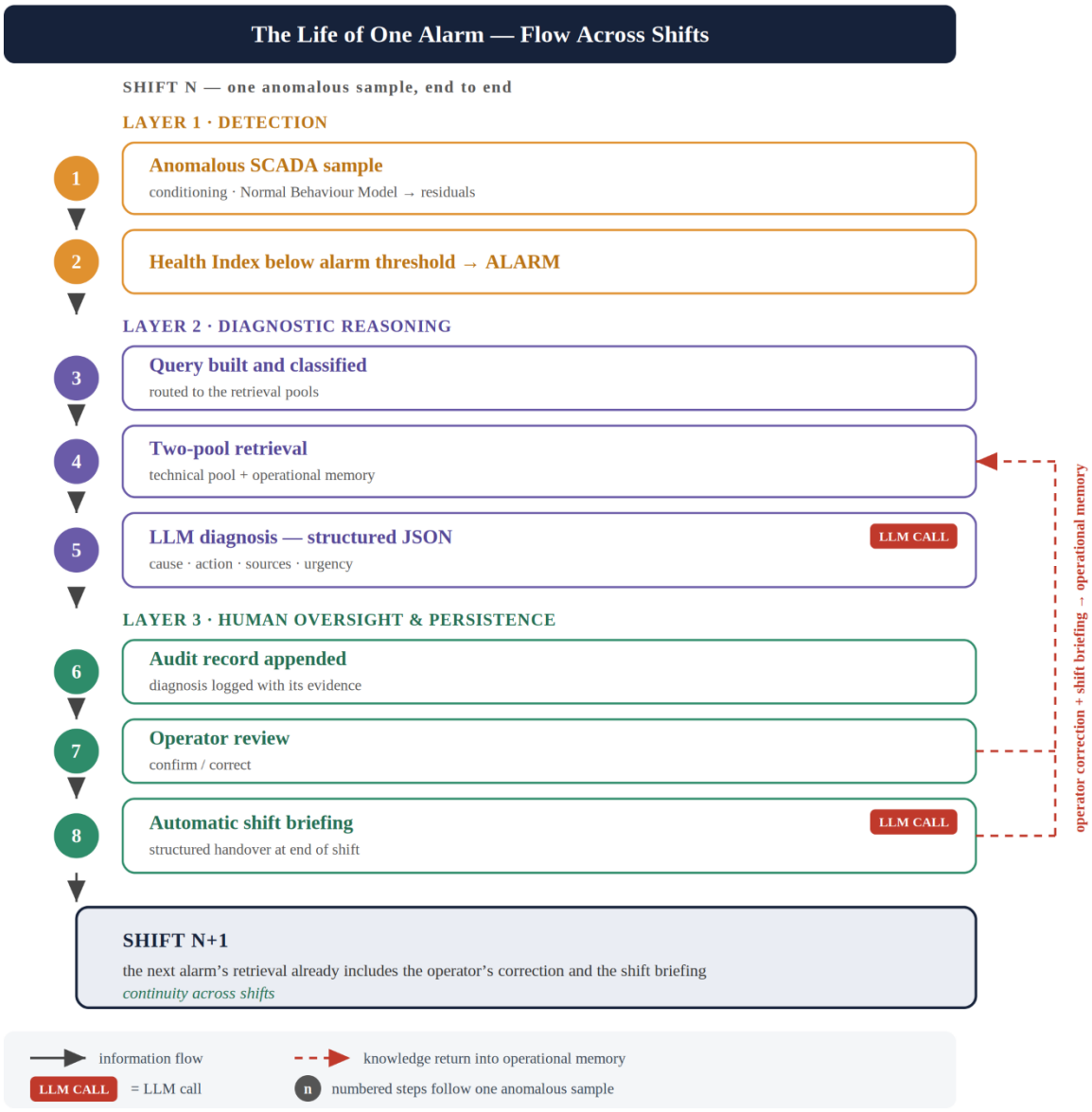

Figure 3.2. *The life of one alarm (nominal flow).*

The downward flow has two interfaces. The first, between detection and reasoning, carries the *evidence package*: the conditioned record together with its expected values, residuals, Health Index, asset state, and alarm flag. The interface is deliberately minimal and wholly numerical, evidence, not interpretation, and it is also the activation contract of the reasoning layer, which engages either when the alarm flag is raised or when an operator addresses a query directly. The second interface, between reasoning and oversight, carries the *structured diagnostic record*. A design choice already noted in Section 3.3 takes on its full meaning here: the record crosses the boundary self-sufficient, carrying its diagnostic content, both

urgency values, the retrieval metadata, and the identities of every fragment used, so the audit component persists what it receives without reconstructing anything, the layer that records is never asked to re-derive what the layer that reasoned already knew.

The upward flow is a single interface, and its asymmetry with the downward path is deliberate. Where the downward interfaces are synchronous, every sample submitted for diagnosis yields a structured record, and every record is persisted to the audit trail, including those produced under error conditions, the return path is asynchronous and gated: it is fed only by acts of human judgement, review and shift handover; its artefacts enter only the operational-memory pool, never the curated strata; and they reach future diagnoses only through the bounded, threshold-gated quota of Section 3.3. Knowledge flows back into the system on human initiative alone, and under explicit admission control.

The temporal dimension completes the picture, and it is why the framework was called a loop rather than a pipeline back in Section 3.1. The loop closes at two timescales: within the shift, where a review becomes retrievable the moment it is recorded, and across shifts, where the briefing carries the synthesis of an entire audit window over the handover boundary and carries forward, cumulatively, the set of anomalies still open, so a deviation first seen several shifts earlier stays visible at every handover until it is resolved. In *Figure 3.2*, the correction the operator records at step 9 is already part of the evidence available at step 11, something no feed-forward arrangement of the same components would show.

One last property deserves stating outright. The authoritative, persistent state of the framework is human-readable in its entirety: the knowledge base is a collection of plain-text documents, the audit trail an open tabular file, the briefing history a readable archive. Two qualifications keep the claim precise. The binary embedding caches that travel with these artefacts are derived, non-authoritative representations, recomputable at any time from the textual sources they encode; and the volatile state of a running session, the in-memory fragment collections, embedding matrices, and classification centroids, is itself rebuilt from the persistent artefacts at start-up, holding nothing that outlives them. Within those bounds, what the system knows and what it has done can be inspected without specialised tooling. *Table 3.1* summarises the seven components, their interfaces, and the engineering decisions that shaped them.

***Table 3.1**. Synthesis of the seven AURA-PHM components: layer membership, input–output interfaces, and key engineering decisions.*

Component	Layer	Main input	Main output	Key engineering decision
Data ingestion and conditioning	1, Ingestion and detection	Raw 10-minute SCADA archive	Conditioned operational records	Productive-regime restriction as a modelling precondition; structural training–test temporal integrity
Normal Behaviour Model	1, Ingestion and detection	Conditioned records	Expected values, per-signal residuals, calibration constants	Independent per-signal regressors for component-level attribution of anomalies
Health assessment and alarm	1, Ingestion and detection	Residuals and calibration constants	Health Index, asset state, alarm flag	Weighted synthesis with fixed normalisation; rule-based alarm trigger
Stratified knowledge base	2, Diagnostic reasoning	Curated sources; runtime briefings and feedback	Attributable fragments in two retrieval pools	Pool separation at the storage level; single shared embedding space
Diagnostic reasoning engine	2, Diagnostic reasoning	Evidence package or operator query; retrieved fragments	Structured diagnostic record with retrieval metadata	Conservative classification fallback; constrained JSON generation; dual urgency
Traceability and audit	3, Human oversight and persistence	Diagnostic record and triggering evidence	Persistent audit trail	Co-location of detection and retrieval evidence; single source of record
Operator interaction and shift continuity	3, Human oversight and persistence	Pending events; operator judgement	Resolved records, feedback artefacts, shift briefings	Human gate on knowledge growth; briefing as a synthesis of the audit trail, with deterministic, cumulative carry-forward of open anomalies across shifts

4. Instantiation on the Wind Turbines Validation Case

This chapter turns the framework of Chapter 3 into a running system, instantiated on a real wind-turbine case built from the EDP open dataset. It first sets out how the move from architecture to implementation is handled, then follows the three layers in the order they were designed: the detection layer as a trained Normal Behaviour Model and its Health Index, the stratified knowledge base and its retrieval pools, the diagnostic reasoning path from query classification to constrained generation, and the oversight and persistence layer as the audit trail, feedback mechanism, and shift briefing. It closes by characterising the system's persistent state and summarising the instantiation that Chapter 5 then validates.

4.1 From framework to running system

The previous chapter described AURA-PHM in domain-general terms: what its three functional layers are responsible for, the seven components that carry that out, and the engineering decisions behind them. This chapter pins that description to a single concrete case. Every parameter left symbolic in Chapter 3, the monitored signals and their weights, the training period, the calibration maxima, the Health Index thresholds, the chunking and retrieval settings, the prompt and the urgency rule, is given its operative value here and traced to the implementation unit that uses it. The line with the next chapter is deliberate, and held throughout: Chapter 4 reports *how the system was instantiated* and with which fixed values, while every quantity the instantiated system went on to *produce*, the blind detection over 2017, the lead time before the certified failure, the urgency distribution, the retrieval-quality evaluation, the end-to-end use cases, belongs to Chapter 5. What is settled here is the running system; what it did is reported there.

The validation case comes from the EDP open SCADA dataset, a wind farm of four onshore turbines (T01, T06, T07 and T11) instrumented with standard ten-minute aggregated channels across the two operating years 2016 and 2017. In keeping with the blind protocol of Section 3.2, 2016 supplies the healthy historical reference from which normal behaviour is characterised, and 2017 is the evaluation stream on which anomalies have to surface from the data alone. The detection layer is instantiated on turbine T06, whose 2017 history holds a certified gearbox-bearing failure, and its generalisation is checked on turbine T01 as a cross-asset control. The empirical stratum of the knowledge base draws on the documented failure history of the same fleet: alongside the T06 gearbox-bearing case the detection layer

targets, it records the generator-bearing failure of T07 and the transformer-fan failure of T01 as retrievable diagnostic precedents, each annotated with its characteristic residual signature. Turbine T11, though part of the same wind farm, brings no documented failure case and enters neither the detection pipeline nor the empirical stratum. Three SCADA temperature channels serve as the monitored target signals, gearbox oil, gearbox bearing, and generator bearing temperature, chosen for the criticality and repair cost of the components they watch, as established in Section 3.2; a fixed set of six operational channels describing the environmental and production state of the asset is the common input to the detection model. Restricting the system to channels already collected for operational purposes, with no extra sensing hardware, follows directly from the accessibility argument of Section 1.2.

The framework is realised as a set of cooperating Python components, run in sequence by a single entry point rather than assembled into a heavyweight application framework. The detection layer sits on an ensemble decision-tree regressor from a standard machine-learning library; the diagnostic reasoning layer joins a semantic retrieval mechanism, whose shared embedding space comes from a hosted embedding model, to a hosted large language model under constrained JSON generation; the persistence layer writes to ordinary, directly inspectable files, a tabular audit trail, a knowledge base of plain-text documents, and a readable briefing history. That minimal, file-based realisation is a deliberate engineering position, not an accident. It keeps the authoritative state of the system human-readable in its entirety, as Section 3.5 requires, and it is the concrete counterpart of the deterministic–probabilistic division of responsibility, in which the model is one bounded component among several and never the substrate the system's state is kept on.

As Section 3.1 flagged, the three layers are functional constructs, not separate software modules: the seven components are spread across a small number of source units, and several units carry logic belonging to more than one architectural concern. Chapter 3 held off on the explicit correspondence between the functional decomposition and its implementation; Table 4.1 supplies it, mapping each of the seven components onto the source unit and the principal functions that realise it, and noting the cross-cutting cases where a single unit serves more than one component. The table is the orientation map for the rest of the chapter, whose sections instantiate the components in architectural order: the ingestion and detection layer in Section 4.2, the stratified knowledge base in Section 4.3, the diagnostic reasoning layer in Section 4.4, and the human oversight and persistence layer in Section 4.5.

One further artefact is named here only to put it outside the running system. The retrieval-quality of the diagnostic layer is assessed, in Chapter 5, by a separate evaluation routine that is no part of the operational pipeline and plays no role at runtime: it reads the instantiated knowledge base and the recorded retrievals offline and scores them against a set of reference questions. Since it neither conditions data, nor detects anomalies, nor produces any diagnosis the system acts on, it is described where its results are reported, at the opening of Chapter 5, and is mentioned here only to draw the boundary of what is being instantiated.

***Table 4.1**. Mapping of the seven AURA-PHM components onto their implementation units.*

Layer	Component	Implementation unit	Principal functions
Ingestion and detection layer	Data ingestion and conditioning	`AURA_PHM_L2_NBM_v3.py`	`carica_csv, preprocessing`
	Normal Behaviour Model	`AURA_PHM_L2_NBM_v3.py` *(training);* `AURA_PHM_L3_Agent_v2.py` *(runtime residuals)*	`addestra_nbm`*;* `calcola_residui` *(runtime residuals)*
	Health assessment and alarm generation	`AURA_PHM_L2_NBM_v3.py` *(batch);* `AURA_PHM_L3_Agent_v2.py` *(runtime)*	`calcola_hi, genera_report_allarmi` *(batch);* `calcola_residui` *(runtime HI / state / alarm)*
Diagnostic reasoning layer	Stratified knowledge base	`AURA_KB_Builder.py` *(source authoring, procedural, no functions);* `AURA_PHM_L3_Agent_v2.py` *(chunking, embedding, indexing)*	`costruisci_kb, _dividi_in_chunk, _calcola_embedding, _costruisci_embeddings_kb` *(all in AURA_PHM_L3_Agent_v2.py)*
	Diagnostic reasoning engine	`AURA_PHM_L3_Agent_v2.py`	`_classifica_query, cerca_kb, genera_diagnosi, _urgenza_da_hi`
Human oversight and persistence layer	Traceability and audit	`AURA_PHM_L3_Agent_v2.py`	`salva_audit, applica_feedback_a_record`
	Operator interaction and shift continuity	`AURA_PHM_L3_Agent_v2.py`	`feedback_hitl, lista_anomalie_aperte_cumulativa, genera_briefing`

Note. File and function names are reproduced verbatim from the implementation. The three functional layers are realised across a small set of cooperating Python units rather than as separate modules, run in sequence by main.py through the AURAPHMSystem class; orchestration is not itself one of the seven functional components. Two cross-cutting cases are worth noting. The Normal Behaviour Model is trained in AURA_PHM_L2_NBM_v3.py, while per-sample residuals at runtime are computed in AURA_PHM_L3_Agent_v2.py (calcola_residui); the same function also produces, inline, the

runtime Health Index, asset state and alarm flag for the diagnosed sample, whose batch forms over the 2017 evaluation stream are calcola_hi and genera_report_allarmi in AURA_PHM_L2_NBM_v3.py. AURA_KB_Builder.py authors the four source documents procedurally and defines no functions of its own; the chunking, embedding and indexing functions attributed to the stratified knowledge base all live in AURA_PHM_L3_Agent_v2.py.

4.2 Instantiating the ingestion and detection layer

The ingestion and detection layer of Section 3.2 is instantiated on turbine T06: its 2016 history supplies the healthy reference, its 2017 stream is the blind evaluation target. Turbine T01 is conditioned identically and run through the same trained models as a cross-asset control. The sections that follow fix the operative parameters of the three components: the conditioning filters and train–test separation (Section 4.2.1), the regression configuration and calibration constants of the Normal Behaviour Model (Section 4.2.2), and the weighting, normalisation, and thresholds of the health assessment (Section 4.2.3). Every value below is fixed before evaluation; nothing is learned from the 2017 stream.

4.2.1 Data ingestion and conditioning

Ingestion takes in the raw SCADA archive of ten-minute aggregated channels and gives it a canonical temporal form. Each record's timestamp is parsed to a uniform UTC representation, and the archive is ordered chronologically before anything else happens, so every downstream operation acts on a single, consistently ordered stream. Signal values are left untouched at this stage: ingestion sets up only the temporal frame that conditioning and detection depend on.

Conditioning then narrows the ordered archive, one asset at a time, to the records on which normal-behaviour modelling means something, through the three operations set out in Section 3.2.1. The first restricts the data to the productive operating envelope. A record is kept only when ambient wind speed sits between the asset's cut-in and cut-out limits, 3.0 and 25.0 m/s, and active power output is at least 10 kW. The wind window is therefore the certified operating range of the turbine, not an arbitrary band, while the power floor drops records where the turbine is at standstill, idling, or otherwise unproductive despite admissible wind. Those are regimes where the temperature response to operating conditions does not hold, and letting them in would contaminate both training and residual evaluation. The second enforces completeness: any record missing a value among the six input features or the three monitored targets is discarded, so every record that reaches the model carries the full set of variables it needs. The third separates training from evaluation in time. The training

stream comes only from the 2016 archive and the evaluation stream from the 2017 archive; the two are loaded as independent inputs, and the model is fitted on the conditioned 2016 records alone. The separation is enforced, not assumed: before fitting, the training procedure asserts that no training record carries a timestamp past 2016 and halts otherwise, so no evaluation-period record can reach the characterisation of normality.

Two properties of this instantiation matter. First, the criteria are a modelling precondition, not generic cleaning: each excluded record belongs to a regime the model is not meant to represent, and applying identical filters to the training asset, the evaluation stream, and the cross-asset control makes the three commensurable by construction. Second, conditioning stays confined to selection and completeness, with no smoothing, imputation, or rescaling, so the downstream residuals reflect deviations of the measured signal itself rather than of a transformed surrogate, in line with the evidence-preservation principle of Section 3.2.3.

4.2.2 Training and the Normal Behaviour Model

The Normal Behaviour Model is realised as three independent regressors, one per monitored target, rather than a single multi-output model: each learns the expected value of its own target temperature from the six operational input features. Each regressor is a random forest of one hundred trees with a fixed random seed, so training is exactly reproducible, and unbounded tree depth. The three models are fitted on the conditioned 2016 records of turbine T06 alone, 33,078 samples after conditioning, under the temporal-separation guarantee of Section 4.2.1.

The six input features describe the environmental and production state of the asset: ambient wind speed, rotor speed, ambient temperature, nacelle temperature, blade pitch angle, and active power output. They are common to all three models, and none of the three monitored temperatures is used to predict another. For each record a model returns an expected temperature, and the residual, the signed difference between the measured and the expected value, is the elementary unit of evidence on which all downstream assessment is built. Its sign is kept, since an over-temperature and an under-temperature carry different diagnostic meaning.

Training freezes, for each target, the constants the runtime never recomputes. Two feed the health assessment of Section 4.2.3 directly. The first is the aggregation weight, which sets the target's contribution to the combined Health Index; the weights are fixed at 0.40, 0.35, and 0.25 for the gearbox-oil, gearbox-bearing, and generator-bearing temperatures

respectively, and sum to unity. The second is the normalisation denominator, defined per target as the largest absolute residual seen in-sample on 2016, against which a runtime residual is later scaled before it contributes to the index.

Training also records, for each target, the in-sample fit on the 2016 reference period: the mean absolute error and the coefficient of determination, both computed on the same records used to fit the model. These say how closely each model reproduces normal behaviour over the reference period; they are not measures of generalisation, and the model's behaviour on the unseen 2017 stream is reported, as a result, in Chapter 5. The normalisation denominators are in-sample 2016 quantities in the same sense. The three weights, the three denominators, and the two fit metrics per target make up the fixed calibration of the Normal Behaviour Model and are reported in *Table 4.2*.

***Table 4.2**. Fixed calibration of the Normal Behaviour Model (per target).*

Target	Weight	res_max (°C)	MAE 2016 in-sample (°C)	R^2 2016 in-sample
Gear_Oil_Temp_Avg	0.40	5.89	0.590	0.9718
Gear_Bear_Temp_Avg	0.35	7.88	0.443	0.9879
Gen_Bear_Temp_Avg	0.25	15.02	1.133	0.9852

Note. The two fit metrics are in-sample on the 33,078 conditioned 2016 records of turbine T06; res_max is the per-target normalisation denominator used by the Health Index (Section 4.2.3). Training uses a fixed random seed and is exactly reproducible.

4.2.3 The Health Index and its thresholds

The health assessment fuses the three per-signal residuals into the scalar Health Index defined in Section 3.2.3. For the validation case the index is instantiated with the calibration of *Table 4.2*: each residual's absolute value is divided by its fixed calibration maximum, the ratio is capped at one, the three capped values are combined with the weights 0.40, 0.35, and 0.25, and the weighted sum is subtracted from one. The index is therefore bounded in [0, 1], each contribution capped at its weight, the weights summing to one, with one denoting

agreement with learned normal behaviour and falling values signalling growing deviation. The same computation runs in batch over the evaluation stream and per record at diagnosis time, so the figure an operator sees is the figure the diagnostic layer reasons over.

Two fixed thresholds cut the index into the three states of Section 3.2.3. For the validation case they are set at 0.60 and 0.40: a record is normal at or above 0.60, in attention between 0.40 and 0.60, and in alarm below 0.40. Crossing the alarm threshold raises the binary flag that activates the diagnostic reasoning layer. Both thresholds are fixed before evaluation and applied identically to the training asset, the 2017 evaluation stream, and the cross-asset control.

These thresholds are design parameters of the system, set on the dimensionless Health Index and calibrated to its behaviour on the EDP data; they are not normative prescriptions. In particular they must not be confused with the absolute shutdown limits of IEC 61400-4, for the bulk lubricant, the 85 °C ten-minute-average limit of Clause 6.6.7.2, which are defined on absolute temperatures, set well above the normal operating range, and act only once a measured temperature reaches that level. The Health Index threshold, by contrast, fires on a normalised departure from expected behaviour while the absolute temperature is still inside its normal band. The two work on different quantities and at different stages, and the gap between them is precisely the early-warning margin the residual formulation is designed to open.

Beyond the scalar, each assessed record is tagged with its most anomalous component, the signal whose residual is largest relative to its own calibrated maximum. This selection is unweighted, and the difference from the index is deliberate: the index applies the criticality weights to summarise overall health, whereas the most-anomalous-component tag names the signal that deviates most in its own terms, whatever its judged criticality. In keeping with the evidence-preservation decision of Section 3.2.3, the per-signal expected values and residuals are kept alongside the Health Index, the state, the alarm flag, and this tag, so what crosses into the diagnostic layer is numerical evidence rather than interpretation.

4.3 Populating the stratified knowledge base

The knowledge base of Section 3.3.1 is instantiated as four curated technical sources and one initially empty operational-memory pool. This section fixes the four design-time strata and their anchoring to the standard (Section 4.3.1), the segmentation, embedding, and

routing that turn the sources into a retrievable pool (Section 4.3.2), and the runtime operational memory that forms the second pool (Section 4.3.3).

4.3.1 The four technical strata

The curated technical pool is instantiated as four plain-text documents, one per design-time stratum, each a distinct and individually identified source. The normative-technical stratum is a gearbox failure-mode analysis that keeps two layers explicitly apart. Its certified layer carries the failure-mode taxonomy and the absolute safety limits of IEC 61400-4; a separate methodological layer, marked in the document itself as a design choice of this work rather than part of the standard, ties residual patterns to the failure modes of that taxonomy. The empirical stratum is an archive of documented EDP failure cases, the gearbox failure on the detection asset, the generator-bearing and transformer cases on the other turbines, each recorded with its residual signature, alongside field-reliability figures that set the economic context. The operational-contextual stratum describes the normal operating envelope and the seasonal behaviour of the monitored turbines. The decision-support stratum encodes the mapping from diagnostic findings and Health-Index levels to recommended actions and their urgency. As Section 3.3.1 established, the stratification is a curation by source identity rather than a structural partition: the four share a single pool, and the only boundary enforced by construction is the one dividing this curated pool from the runtime operational memory of Section 4.3.3.

The anchoring to IEC 61400-4 is deliberately partial and explicit. Three of the four strata rest on the standard: the normative-technical stratum carries its failure-mode taxonomy and its absolute limits, the operational-contextual stratum cites the clauses that give the monitored temperatures their physical meaning, and the decision-support stratum ties its verification actions, lubricant analysis, magnetic-particle and endoscopic inspection, to the standard's service and monitoring requirements. The empirical stratum instead rests on documented field evidence rather than on the standard. Throughout, the documents keep what is certified apart from what is a design choice of the system: where a statement follows from the standard it is attributed to it, and where it is an engineering decision, the residual-to-failure-mode association, the urgency timings, the Health-Index levels, the documents say so. This is the instantiation, at the level of the sources themselves, of the principle that certified and operational knowledge hold different epistemic status.

The sharpest case of that separation is the thermal limits. The normative-technical stratum records the absolute shutdown limits of IEC 61400-4, for the bulk lubricant, the ten-minute-

average limit of 85 °C above which the standard requires the turbine to be shut down (Clause 6.6.7.2). These are defined on absolute temperatures and sit well above the normal operating range, where the gearbox oil runs near 50 °C. They are therefore categorically distinct from the data-driven Health-Index thresholds of Section 4.2.3, which act on a normalised deviation from expected behaviour and fire far earlier: a residual of a few degrees does not mean the oil is near its absolute limit, only that it is warmer than expected for the current regime. The knowledge base keeps both, labelled as different objects, so the reasoning layer can cite the early-warning residual evidence and the absolute safety limit without conflating them, which is what stops a diagnosis from dressing an early warning up as the breach of a certified limit.

Table 4.3. *The five strata of the knowledge base, organised in two retrieval pools.*

Stratum	What it encodes (source)	Anchored to IEC 61400-4?
CURATED TECHNICAL POOL · *populated at design time, evolves only through manual curation*		
Normative-technical	A gearbox failure-mode analysis: its certified layer carries the failure-mode taxonomy and the absolute safety limits of IEC 61400-4; a separate methodological layer, flagged as a design choice, links residual patterns to those failure modes. (DOC1)	**Yes**
Empirical	An archive of documented EDP failure cases: gearbox, generator-bearing and transformer, each with its residual signature, alongside field-reliability figures that set the economic context. (DOC2)	**No**
Operational-contextual	The normal operating envelope and seasonal behaviour of the monitored turbines, citing the clauses that give the monitored temperatures their physical meaning. (DOC3)	**Yes**
Decision-support	The mapping from diagnostic findings and Health-Index levels to recommended actions and their urgency; its verification actions (lubricant analysis, inspection) are tied to the standard's service and monitoring requirements, while the Health-Index levels and urgency timings are design choices calibrated on the operational data, not IEC prescriptions. (DOC4)	**Yes**
DYNAMIC OPERATIONAL-MEMORY POOL · *empty at deployment, grows autonomously at runtime*		
Operational-memory	Shift briefings and operator feedback, re-indexed as they are produced, the system's own accumulated operating experience rather than curated knowledge. (runtime artefacts)	**No**

4.3.2 Segmentation embedding and pool construction

Each source is segmented into overlapping fixed-length fragments. A sliding window of six hundred characters advances in steps of five hundred, so consecutive fragments share one

hundred characters and local context survives their boundaries; a trailing fragment shorter than a minimum length is dropped rather than kept as a stub. Every fragment records the identity of its source and its sequential position within it, the provenance the audit trail later leans on, and the reason the unit of retrieval is itself attributable.

Pool membership is assigned from the source's name rather than inferred at query time. A declared set of technical prefixes, the document family together with standards identifiers such as IEC and ISO, routes a source to the curated technical pool, while the briefing and feedback prefixes route runtime artefacts to the operational-memory pool; a source whose prefix matches neither goes to the technical pool with a warning, a conservative fallback. The two pools are therefore held as physically distinct fragment collections, each with its own embedding matrix, and the epistemic boundary of Section 3.3.1 is enforced at the storage level rather than by a filter a retrieval fault could slip past.

Each fragment is projected into a semantic space of 1536 dimensions by a single hosted embedding model, and every resulting vector is L2-normalised, so similarity between a query and a fragment comes down to an inner product. The same model embeds the curated fragments, the runtime artefacts, and the queries alike: this shared space is what lets knowledge generated at runtime be retrieved by exactly the mechanism that retrieves curated knowledge, the precondition for the loop the human-oversight layer closes.

Because each embedding call carries an API cost, the vectors are treated as a derived cache rather than primary state. Each pool's embedding matrix is computed once and persisted to its own file, and on later starts it is loaded rather than recomputed, so the cost is paid once per revision of the knowledge base, not once per query. The cache is not trusted blindly. At load, the system checks each pool's fragment count against the number of rows in its embedding matrix, recomputing or warning on any mismatch, so a stale cache cannot quietly pair a fragment with the wrong vector before queries are accepted.

4.3.3 The runtime operational memory

The operational memory is the second retrieval pool, and unlike the four technical strata it is not authored before deployment. At start-up it is empty: the operational fragment collection holds nothing and the operational embedding matrix does not yet exist. It fills only at runtime, from the artefacts the human-oversight layer produces, the shift briefings and the operator feedback instantiated in Section 4.5, so its contents are the system's own accumulated operating experience rather than curated knowledge.

When such an artefact is produced, it is given a source name carrying an operational prefix, briefing or feedback, and then handled by exactly the machinery of Section 4.3.2: segmented by the same procedure, routed by its prefix into the operational pool, embedded by the same model. Its vectors are appended to the operational embedding matrix, which is initialised if the pool was empty and extended otherwise, and the pool's cache is rewritten to disk. The artefact becomes retrievable within the same running session, with no restart, and the fragment count and the embedding matrix stay aligned as the pool grows.

Across sessions the operational memory persists through the artefacts themselves. The briefing history and the feedback records are stored as files and reloaded at the next start-up, while their embeddings follow the cache discipline of Section 4.3.2, restored from the operational cache when it is coherent with the reloaded fragments, and recomputed only when the artefacts have changed. The pool's contents therefore come from the system's own operating record rather than from a curated source. Because every fragment, curated or runtime-generated, lives in the same embedding space and is retrieved by the same mechanism, knowledge written during one shift becomes available to the next through ordinary retrieval, with no special path. This is the closed loop of Section 3.3.1 made concrete: the system's own outputs re-enter its knowledge base as first-class, retrievable evidence, which is what equips the operational memory for the shift-to-shift continuity developed in Section 4.5.

4.4 Instantiating the diagnostic reasoning layer

The diagnostic reasoning layer of Section 3.3.2 governs the path from an activating anomaly, or an operator's question, to a structured diagnostic record. Its engine runs in four stages: query understanding, pool routing and retrieval, constrained generation, and deterministic post-processing. The first two are instantiated in Sections 4.4.1 and 4.4.2; the last two, run in immediate succession, in Section 4.4.3. The division of labour set in Chapter 3 holds: inside the engine the large language model is confined to a single generative call, while classification, routing, the authoritative urgency, and the recording of the result stay deterministic, so the probabilistic component is bounded on both sides by machinery whose behaviour is fixed.

4.4.1 The semantic query classifier

Before any retrieval, the engine classifies the incoming query, an operator's free-text question, or the implicit query formed when a detected anomaly activates the layer, into one of three categories: technical, operational, or diagnostic. The category decides which knowledge the next stage retrieves.

The classifier is an example-based heuristic, not a trained model. Each category is represented by a centroid: the mean of the embeddings of a small fixed set of prototype queries, eight per category, L2-normalised in the same 1536-dimensional space as the knowledge base. The technical prototypes ask about standards, limits, and failure modes; the operational ones ask what happened in previous shifts; the diagnostic ones state a live anomaly and ask what to do. A query is classified by embedding it and taking the category whose centroid is closest in cosine similarity.

Because the heuristic is lightweight, its dependability rests on two explicit guards. The best-matching similarity has to clear an absolute confidence of 0.30 and lead the second-best category by at least 0.05; if either guard fails, or the query cannot be embedded, the engine falls back to the diagnostic category, the broadest retrieval configuration of Section 4.4.2. Classification uncertainty therefore pushes the system toward more evidence, never toward less: an ambiguous query gets the full diagnostic treatment rather than being narrowed, perhaps wrongly, to a single pool.

4.4.2 Pool routing and retrieval

The category the classifier assigns selects the pool. A technical query is answered from the technical pool, an operational query from the operational pool, each returning its three best-matching fragments. A diagnostic query is the only configuration that draws on both: it retrieves the same three technical fragments and, on top, at most one operational fragment. Falling back to it therefore widens the evidence base rather than confining an uncertain query to a single pool, and leaves the curated technical core untouched.

The lone operational fragment is admitted only if its similarity to the query clears an absolute bar of 0.38, a threshold on the similarity itself, not on a score rescaled against the pool, so weakly related memory is shut out rather than let in for merely being the pool's best. The smaller operational quota and this admission bar encode one decision: runtime memory may inform a diagnosis, but must not crowd out the curated technical knowledge that anchors it.

Retrieval is semantic by default, cosine similarity over the cached embeddings of Section 4.3.2, with a lexical mode and a hybrid mode (the two rankings combined with fixed weights, 0.6 and 0.4, after normalisation) available as the baselines compared in Chapter 5. In every mode a redundancy control closes the stage: a candidate is dropped if it repeats, or sits immediately next to, a fragment already chosen from the same source, so the overlap built in at segmentation cannot fill the context with near-duplicates.

Whatever is selected carries its provenance onward. Each fragment keeps its source, its position within that source, and its retrieval score, and these per-fragment records are written to the audit trail of Section 4.5, not just summarised, so a diagnosis can be traced back to the exact fragments that informed it. One boundary case is handled rather than left to fail: an operational query posed before the operational memory holds anything, the cold-start condition of Section 4.3.3, searches the combined collection instead of returning nothing.

4.4.3 Constrained generation and determinist post processing

The engine assembles a single prompt from two ingredients: the numerical evidence package, the Health Index, the asset state, the most anomalous component, and the per-signal residuals against their expected values, and the fragments retrieved for the query, each labelled with its source. One element of the evidence is fixed deterministically before the model sees it. The sign of the most anomalous component's residual is computed and written into the prompt: a positive residual means a temperature above expectation, consistent with developing overheating, while a negative one points to a transient, a partially stopped machine, or sensor drift rather than a fault. The prompt tells the model to follow this reading, so the residual's absolute magnitude, which drives detection severity, is not mistaken for its physical cause.

The prompt goes to GPT-4o-mini once, at a temperature of 0.1, in constrained JSON mode. The model has to return a fixed schema: a probable diagnosis, a root cause, the residual pattern it reports, which the prompt tells it to keep consistent with the sign supplied, an urgency, a recommended action, the principal source relied on, and a declared limitation, a field in which the model states what its assessment cannot establish.

Two properties of this stage need stating precisely. Constrained generation guarantees the structural validity of the output by construction, doing away with any fragile post-hoc parsing; but structural validity says nothing about the quality of the reasoning inside the structure, which turns on the quality of the retrieved context, the dependency that drives the

retrieval-focused evaluation of Chapter 5. The near-zero temperature, for its part, reduces variation between calls without making the step deterministic: the framework makes no such claim, and pursues reproducibility through recording rather than through an assumption of determinism.

The model's urgency is not the operative one. Independently, the engine derives the operational urgency from the Health Index through a fixed rule. Applied to the alarming samples that reach it, those below the 0.40 alarm threshold, the rule sorts them into four windows, from immediate through 24 hours and 48–72 hours to seven days, the lower the index the more urgent (*Figure 4.1*). Of the rule's three internal boundaries, only the critical one at 0.20 is shared with the decision-support stratum; the cuts at 0.27 and 0.34 are the system's own calibration, **the median and the 75th percentile of the Health Index over the 176 alarm records of its calibration audit**, chosen to keep urgency from collapsing into a single class. The rule does not otherwise transcribe that document's action levels, it borrows the critical anchor and is independent elsewhere, and the document itself declares its Health-Index thresholds to be design choices calibrated on the operational data, not normative prescriptions. This deterministic urgency is authoritative; the model's own urgency is kept beside it as an observational value, so the two can be compared rather than one silently replacing the other.

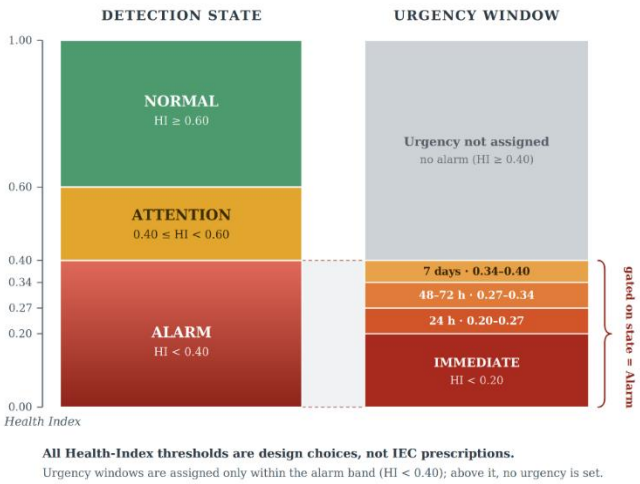

Figure 4.1 . *The Health-Index threshold scheme.*

The record that leaves the engine therefore carries, with the diagnosis, both urgency values, the retrieval query and mode, and the per-fragment provenance of Section 4.4.2, everything the human-oversight layer needs to reconstruct how the diagnosis was produced. The structure survives even an API failure: the deterministic urgency is still computed and the record still leaves the engine well-formed, so a downstream fault cannot strip a diagnosis of its operational urgency.

4.5 Instantiating the human oversight and persistence layer

The third layer does two things at once: it opens every diagnosis to human review, and it turns what the system and its operators decide into knowledge the system keeps. It is instantiated as three persistence channels. The audit trail (Section 4.5.1) records each diagnosis together with the evidence behind it; the operator-feedback mechanism (Section 4.5.2) lets a human confirm or correct that record; and the shift briefing (Section 4.5.3) carries the result forward to the next shift. The loop anticipated in Chapter 3 closes here: a corrected diagnosis, and each shift briefing, is written back into the operational memory as a new artefact, so the system's reviewed experience becomes retrievable evidence for later diagnoses, the shift-to-shift continuity the framework is built around.

4.5.1 The audit trail

The first channel is an audit trail: every diagnosis the engine produces is appended, as a single row, to a persistent file, which therefore grows by one entry per diagnosed sample and holds a complete, ordered history of what the system decided and on what basis. What the engine writes is never rewritten afterwards; only two fields of each row, reserved for human review, are filled in later, in place, when an operator examines the record (Section 4.5.2).

Each row holds seventeen fields. They identify the event (the turbine, the timestamp of the SCADA sample, and the timestamp of the diagnosis); they carry the detection numbers (the Health Index, the state, the most anomalous component, and the magnitude of its residual); they carry the diagnosis itself (the probable diagnosis, the recommended action, and the principal source); and they carry both urgency values side by side, the deterministic one that governs the response and the model's observational one. Two further fields record how the evidence was gathered, the retrieval mode and the per-fragment provenance, which for each fragment actually used keeps its source, its position within that source, and its retrieval score, and one logs the token cost of the generation.

It is this provenance that makes a diagnosis auditable. What the trail preserves is the retrieval mode and pointers to the exact fragments behind each diagnosis, their source, position, and score, not the verbatim query the engine assembled at runtime, and not the fragment text. Identifying the evidence behind a diagnosis therefore means resolving those pointers against the knowledge base. The trail also keeps the magnitude of the most anomalous component's residual but not its sign; the sign that drives the diagnostic pattern of Section 4.4.3 is reflected in the diagnosis itself and stays recomputable from the SCADA sample, rather than recorded as a field. The link the row records is between a diagnosis as a whole and its evidence set, attribution at the level of the output, not a claim-by-claim mapping. The two remaining fields carry the human review: a status the engine initialises to "pending", and an operator note left empty until the step of Section 4.5.2 fills them. Together they make the trail the substrate of the framework's traceability, any past diagnosis can be revisited and traced to the evidence behind it and to how its urgency was set.

Table 4.4. *Fields of the audit-trail record. Every diagnosis appends one row; the engine writes all fields except the two review fields, which an operator completes in place.*

Field	What it records
EVENT IDENTIFICATION	
Diagnosis timestamp	When the diagnosis was produced
Turbine	The asset diagnosed
SCADA timestamp	Timestamp of the source SCADA sample
DETECTION	
Health Index	The health index computed for the sample
State	Normal, attention or alarm
Most anomalous component	The component with the largest normalised residual
Residual magnitude	The absolute residual of that component
DIAGNOSIS	
Probable diagnosis	The model's probable diagnosis
Recommended action	The recommended action
Principal source	The main knowledge source relied on
URGENCY	
Urgency (deterministic)	Authoritative, derived from the Health Index
Urgency (model)	The model's own, recorded as observational
PRODUCTION METADATA	

Field	What it records
Retrieval mode	Semantic, lexical or hybrid
Per-fragment provenance	For each fragment used: its source, position and retrieval score
Token cost	Tokens consumed by the generation
HUMAN REVIEW	
Review status	PENDING, then CONFERMATO or CORRETTO
Operator note	Free-text note, empty until the record is reviewed

4.5.2 The operator feedback mechanism

When a diagnosis is produced it enters its record as “PENDING”. The operator, at the alarm itself or when reviewing the shift, returns one of two outcomes: the diagnosis is confirmed, or it is corrected. Either outcome updates the record in place, setting its status and attaching any free-text note; a correction also carries the operator’s account of the real cause.

The two outcomes differ in what they add to the knowledge base, and the difference is deliberate. A correction brings something the system did not have, the real cause, and is written out as a new operational artefact: a short document stating that the case was corrected, the real cause, and any note, which is segmented, embedded, and appended to the operational pool by the machinery of Section 4.3.3, and so becomes retrievable for later diagnoses. A confirmation brings nothing of the kind: the diagnosis already stood, so there is no new knowledge to store and the operational pool does not grow. A guard keeps even corrections from piling up redundantly, a correction is indexed only when it is new or its content has changed.

A confirmation is not inert, though. An anomaly is open from the moment its alarm is raised and stays open until a correction closes it; what a confirmation does is record that the operator has judged it real, relabelling it from pending to confirmed without closing it. That recorded status is what the continuity mechanism of Section 4.5.3 carries forward: the anomaly stays on the open list, and so reappears in every later shift briefing, until a correction on the same component closes it. The two outcomes thus act on two different stores, a correction writes to the retrievable knowledge pool and, by closing the anomaly, also drops it from the open list, whereas a confirmation writes only to the audit record, leaving the pool untouched and the anomaly open. This is where an operator’s judgement becomes part of the system, as retrievable knowledge when it corrects, and as recorded status when it confirms, without retraining the model or moving any threshold.

4.5.3 The shift briefing and cross-shift continuity

The third channel is the shift briefing, and it is where the continuity the framework is built around becomes real. At a fixed cadence, every forty-eight records of the replay stream, nominally eight hours at ten minutes per record, matching a typical shift rotation, the system produces a structured briefing addressed to the next shift's technician.

The briefing has a deliberate division of labour between what is computed and what is written. Its backbone is a deterministic list of open anomalies, assembled straight from the audit: for each turbine and component, an anomaly counts as open if it carries an alarm more recent than its last correction, and it is labelled confirmed if the operator confirmed it and pending otherwise. This list is cumulative across all shifts and sits at the head of the briefing, and the language model is told explicitly neither to reproduce nor to invent it. This is where a confirmed anomaly persists: it stays on the open list, and therefore in every later briefing, until a correction on the same component closes it. Around that backbone, GPT-4o-mini writes the narrative, the current situation, the priority actions, the trend against previous shifts, and the operator's notes, into a fixed structure, at a temperature of 0.2, given the shift's events and the two most recent briefings as context.

Each briefing is then persisted three ways: as a readable file, as an entry in a durable briefing history, and as fragments re-indexed into the operational pool. At the next start-up the stored history is reloaded into the pool, so what one shift recorded becomes retrievable evidence for the diagnoses of the next. The most safety-relevant content travels by the most reliable path: the open-anomaly list reaches the following shift by deterministic construction, independent of the model and of whether retrieval later surfaces the briefing, while the narrative and the retrievable history add context on top. This is the shift-to-shift knowledge continuity the framework set out to provide, made concrete.

4.6 Persistent state and reconstruction

The state the running system depends on is spread across the three layers instantiated above, the Normal Behaviour Model in the detection layer, the stratified knowledge base in the reasoning layer, and the audit trail, feedback artefacts, and briefing history in the oversight layer, yet it resolves into two kinds of very different standing. Some of it is irreplaceable: it was authored, supplied as data, or accumulated as a record of what the system did, and nothing could regenerate it. The rest is computed from that first kind and can be rebuilt at

any time, so its on-disk form is a cache or an export rather than a source of truth. Drawing this line is what makes the system's state well-defined; it fixes exactly what a restart must preserve and what it may safely regenerate.

The irreplaceable state has two origins. The inputs are the curated source documents of Section 4.3 and the SCADA datasets, the 2016 training records and the 2017 evaluation stream. The records are the three artefacts the system builds up as it runs: the audit trail, to which each diagnosis appends a row, the operator-feedback files written whenever a diagnosis is corrected, and the briefing history that captures each shift handover. Inputs and records together are the whole of what no recomputation could recover, and they are exactly what the persistent stores must protect.

Everything else is computed from that state. The Normal Behaviour Model is retrained from the 2016 data at each launch; it is also written to a single archive, but that archive serves inspection and portability rather than reuse, the running model is always the freshly trained one, not a file reloaded from disk. The embedding matrices for the knowledge base and its two pools, and the centroids of the query classifier, are functions of the documents and the operational memory, cached to binary files only to spare their recomputation at every launch. None of these is authoritative.

The computed files that get reused are guarded rather than trusted. An embedding cache is loaded only when it exists and its size still matches the number of chunks it should cover; the classifier centroids only when their array shape matches the expected one. A mismatch in size, the kind a knowledge base that has grown or shrunk would leave behind, is caught and the file recomputed instead of used. The guard is on extent, not content, though: a cache left by a corpus that changed without changing size, the embedding model swapped, or chunk text edited in place, slips through, and has to be cleared by a forced recomputation. A derived file can, in any case, be deleted without consequence.

Start-up is the regeneration of the computed state from the irreplaceable one. The model is retrained; the knowledge base is rebuilt by re-reading the documents in its folder, the curated sources and the feedback files written there during operation, while the briefing history is reloaded from its own record and re-embedded into the operational pool; the cross-shift continuity of Section 4.5.3 is thereby restored across a restart rather than lost. The embedding caches and centroids are then loaded or, where stale, recomputed. The system holds nothing

computed that it cannot regenerate: it can be rebuilt in full from its inputs and its accumulated records alone. *Table 4.5* sets this out element by element.

The instantiated system is driven through a single interactive menu. Its tools cover single-sample and top-alarm batch diagnosis, an operator chatbot, end-of-shift briefing generation, and inspection of the audit trail; operator feedback is collected as a sub-step of each diagnosis rather than as a separate tool. One tool is a chronological replay that streams a chosen interval of the 2017 evaluation data through the full pipeline, producing diagnoses, feedback handling, and briefings as in operation; it is this replay that carries the validation of Chapter 5.

Table 4.5. *The system's persistent state. Irreplaceable elements are authored, supplied or accumulated and cannot be regenerated; computed elements are rebuilt from them at start-up, so their on-disk form is a cache or an export rather than a source of truth.*

State element	On disk as	Standing	Handling at start-up
IRREPLACEABLE · *authored, supplied or accumulated, cannot be regenerated*			
Curated knowledge documents	text files	authored input	Re-read and re-chunked into the technical pool.
SCADA datasets (2016, 2017)	CSV	supplied input	Re-read; the training records feed the model, the evaluation stream the replay.
Audit trail	CSV	accumulated record	Persists on disk; not re-read at start-up, appended per diagnosis (review fields filled in place) and read on demand during operation, e.g. for open-anomaly tracking in briefings.
Operator feedback	text files	accumulated record	Re-read with the documents and re-embedded into the operational pool.
Briefing history	JSON	accumulated record	Reloaded from its record and re-embedded into the operational pool.
COMPUTED · *regenerable from the irreplaceable state*			
Normal Behaviour Model	pickle archive	computed (export)	Retrained from the 2016 data at each launch; the archive is not reloaded.
KB and pool embeddings	binary (.npy)	computed cache	Loaded if their size matches the corpus, otherwise recomputed.
Query-classifier centroids	binary (.npy)	computed cache	Loaded if their shape matches, otherwise recomputed.

5. Experimental Validation

This chapter reports the experimental validation of AURA-PHM. It is organised along the two complementary axes flagged in Chapter 1: a *quantitative* axis, which assesses the detection layer on unseen data and the diagnostic layer through an automated evaluation of its retrieval-augmented answers, and a *qualitative* axis, which examines complete diagnostic cases and the shift-to-shift continuity mechanism. The setup and the evaluation protocol shared by both come first.

5.1 Experimental setup and evaluation protocol

The evaluation rests on the EDP open SCADA dataset. The Normal Behaviour Model is trained only on turbine T06 for 2016, then applied to the same turbine's 2017 stream, which it has never seen. A strict temporal separation holds throughout: no 2017 record, and no information about the certified failure of 17 October 2017, reaches the training stage. After restriction to the productive operating regime, the only conditions under which a normal-behaviour model means anything, the training set comes to 33,078 ten-minute records and the 2017 blind test stream to 36,468. That arrangement makes the 2017 evaluation a genuine out-of-sample test rather than a re-description of the data the model was fitted on.

The diagnostic layer is evaluated with the RAGAS framework of Section 2.6, which scores a retrieval-augmented answer on three dimensions: *faithfulness*, whether the answer is supported by the retrieved context; *answer relevance*, whether it addresses the question posed; and *context relevance*, whether the retrieved context stays focused on that question. The evaluation set is twenty-one curated question–reference pairs spanning the technical and operational strata of the knowledge base. Since both the generation and the automated judging are done by a stochastic language model, every configuration is run three times and reported as the mean and standard deviation across the three, so each figure carries an explicit measure of its own variability rather than resting on a single execution.

Two methodological choices are stated openly here, because both bear on how the diagnostic figures should be read. First, the faithfulness metric was corrected during this work so its value is bounded to [0, 1] and each support verdict is aligned with its corresponding claim; the correction removes an artefact by which the original implementation could return values above unity, and it is applied uniformly, whether it raises or lowers a given score. Second, the answer generator and the automated judge belong to the same model family (GPT-4o-mini). That introduces a well-documented *self-preference bias*, whereby a model tends to

rate its own outputs favourably; the diagnostic scores have to be read with this bias in mind. Both points return in the discussion (Section 5.6.3).

The rest of the chapter follows the two axes. Section 5.2 characterises the detection layer's behaviour on the 2017 stream; Section 5.3 reports the diagnostic layer's RAGAS scores together with an analysis of retrieval quality that reads them; Sections 5.4 and 5.5 present the qualitative validation, through end-to-end diagnostic cases and the shift-continuity demonstration; and Section 5.6 discusses the findings and the limitations that qualify them.

5.2 Detection layer on the 2017 stream

The detection layer was applied to turbine T06's complete 2017 stream, a full year the Normal Behaviour Model had never seen. Over 36,468 productive-regime records it reconstructs the three monitored temperatures with a mean absolute error of about 2 °C, and the resulting Health Index puts 81.3% of records in the normal band, 13.3% in the attention band, and 5.5% in the alarm band, 1,993 alarms across the year. Those alarms cluster around no single event: they are spread over all twelve months, with the deepest Health-Index depressions falling in winter rather than at the certified failure. The detection layer is therefore a high-sensitivity indicator that flags departures from expected behaviour readily, at the cost of a sizeable background alarm rate.

The certified failure of turbine T06, a gearbox bearing failure on 17 October 2017, is examined against this background. In the two weeks before the failure the gearbox thermal residuals sit above their annual baseline, but only weakly: the Gear-Oil and Gear-Bearing residuals average roughly 0.2–0.5 °C above baseline, with a modest weekly peak around 8 October, some nine days before the failure. Over the same window the aggregate Health Index stays near 0.74, inside its normal-to-attention range, and the fraction of alarms in the pre-failure window (about 0.02) is in fact lower than the annual background (about 0.055). The detector does not, then, isolate this failure in advance: no reliable early-warning margin separates the pre-failure window from ordinary annual variability. That fits the physical nature of the fault (a planetary-bearing cage fracture, whose signature is primarily vibrational), which the three SCADA temperatures catch only faintly.

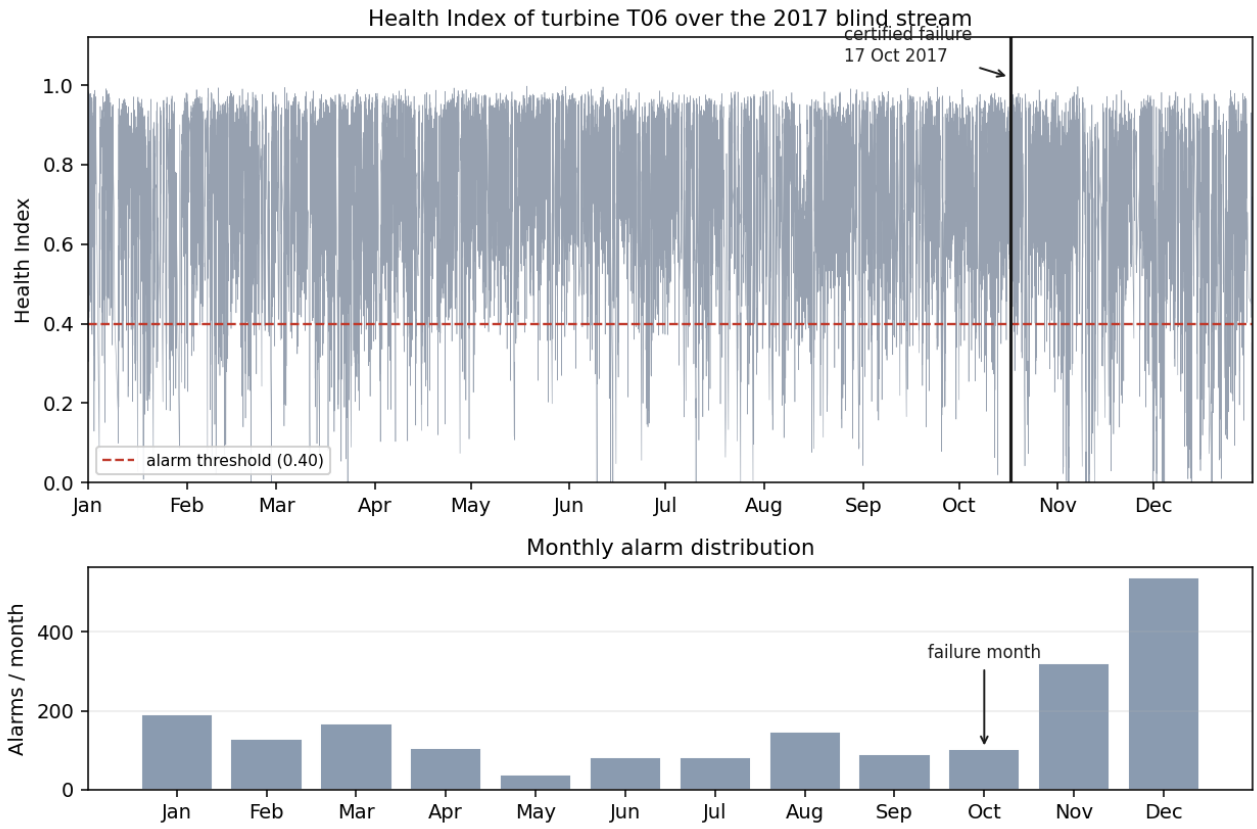

***Figure 5.1**. Health Index of turbine T06 over the 2017 blind stream (top) and monthly alarm distribution (bottom), with the 17 October failure marked.*

A second feature of the year confirms that the aggregate Health Index is not, on its own, a clean degradation indicator. The most sustained Health-Index depression comes in November and December, after the failure, where the monthly mean drops to 0.67 and 0.64 against summer values near 0.77. This depression tracks the seasonal fall in ambient temperature, from roughly 27 °C in August to 14 °C in December, rather than any mechanical event, and is best read as a limitation of a model trained on a single annual cycle: the Normal Behaviour Model does not fully absorb winter operating conditions and reports the resulting departures as anomalies.

Taken together, these observations characterise the detection layer as sensitive but noisy: neither the aggregate Health Index nor a simple alarm count isolates the certified failure from seasonal and operational variability. That characterisation is exactly what motivates the layered architecture. A weak, component-specific thermal signal does exist in the pre-failure window, visible in the targeted Gear-Bearing residual rather than in the aggregate index, and surfacing and attributing such signals is the diagnostic layer's job, while filtering the

seasonal false alarms is the human-oversight layer's. In keeping with Section 3.2, the detection layer is treated here as a diagnostic trigger rather than a predictive instrument: the system's contribution begins where the alarm ends.

5.3 Quantitative evaluation of the diagnostic layer

The diagnostic layer was evaluated with the RAGAS protocol of Section 5.1: twenty-one curated question–reference pairs, each scored for faithfulness, answer relevance, and context relevance, the evaluation repeated three times and reported as the mean and standard deviation across runs *(Table 5.1)*.

***Table 5.1**. RAGAS results over the twenty-one evaluation questions (mean ± std over three runs).*

Metric	Mean ± std	Run 1 · 2 · 3
Faithfulness	0.877 ± 0.006	0.874 · 0.885 · 0.872
Answer relevance	0.707 ± 0.003	0.706 · 0.712 · 0.705
Context relevance	0.159 ± 0.003	0.159 · 0.155 · 0.163
RAGAS score	0.581 ± 0.002	0.580 · 0.584 · 0.580

Faithfulness is high and stable (0.877 ± 0.006): the answers are, in the large majority of cases, supported by the context retrieved for them, and the small standard deviation shows this holds consistently across independent runs. Answer relevance is moderate (0.707 ± 0.003), reflecting answers that address the question while sometimes adding further context-grounded detail. Context relevance is low (0.159 ± 0.003), a value that follows from the interaction between the metric and the retrieval strategy rather than from any weakness in retrieval.

Context relevance estimates the fraction of the retrieved context that is directly pertinent to the question, measured at the level of individual sentences. AURA-PHM retrieves at the coarser level of chunks (three technical and, when admissible, one operational), each of which carries more than the single sentence that answers a given query. A sentence-level metric therefore scores a deliberately broad-chunk strategy low even when the chunks retrieved are the right ones. The per-question distribution *(Figure 5.2)* supports the reading:

the values are uniformly low across all twenty-one questions, the signature of a structural property of the metric rather than of occasional mis-retrieval.

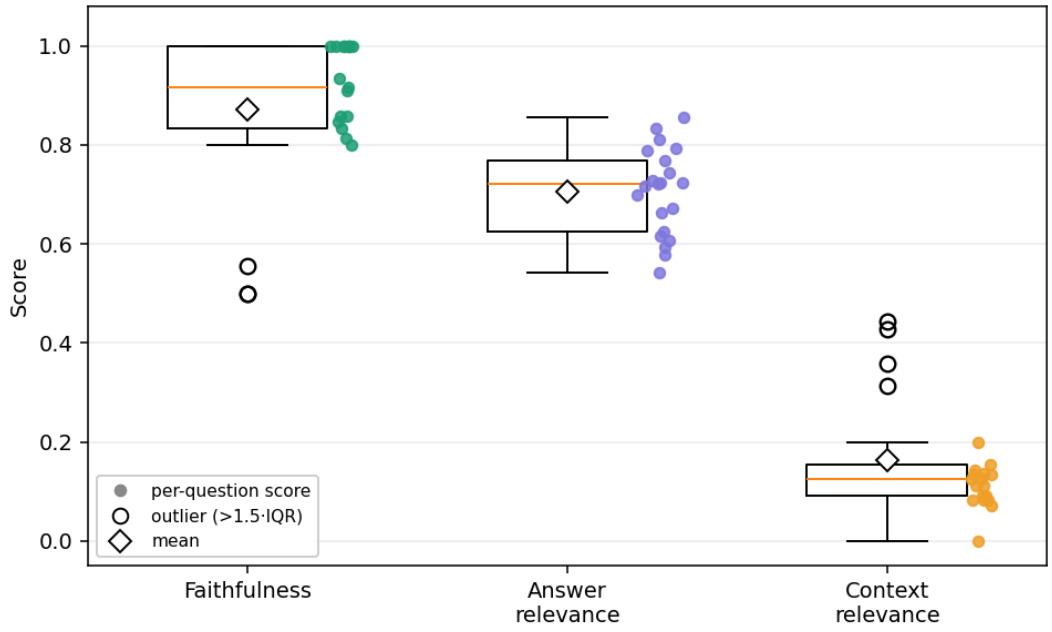

Figure 5.2. *Distribution of the three RAGAS scores across the twenty-one evaluation questions (single representative run).*

A separate look at the retrieval stage confirms that the chunks are well chosen. Across the twenty-one questions the top-ranked technical chunk had a mean cosine similarity of 0.60 to the query (maximum 0.74), with only one question falling below 0.45, a meta-level query on the training methodology, which the knowledge base does not cover. The retrieved chunks matched the expected source documents, and for the historical-case queries the relevant archive entry was generally among the top-ranked results; the operational gate admitted pertinent briefings and feedback while shutting out unrelated content. The picture is one of sound but broad retrieval, exactly the condition under which a sentence-level context-relevance score comes out low despite well-chosen sources. Strong faithfulness and this analysis agree: the reasoning layer is faithful to a context that is itself largely appropriate. The composite RAGAS score (0.581), being the mean of the three terms, is dragged down by the structurally low context-relevance and is less informative than the components taken one by one.

Two qualifications, developed in Section 5.6.3, come with these figures. The automated judge and the answer generator belong to the same model family, so the faithfulness and relevance scores carry a self-preference bias and are best read as an upper-leaning estimate. And the faithfulness value sometimes cited as a RAGAS reference (about 0.96) was obtained by the framework's authors on a general-purpose benchmark, not a fault-diagnosis task; the value reported here is a strong absolute result on a domain-specific, twenty-one-question evaluation, not the outcome of a head-to-head comparison.

5.4 Qualitative validation of the diagnostic layer

The diagnostic layer is validated qualitatively in two steps. Section 5.4.1 shows how an operator's question is routed to the right part of the knowledge base; Section 5.4.2 then follows four real alarms through the full pipeline.

5.4.1 Query classification and pool routing

Before it retrieves anything, the engine sorts each query into one of three categories (technical, operational, or diagnostic), and the category chooses the pool to read from (Section 4.4.1). The classifier is semantic, not lexical: it compares the query's embedding to a centroid of example queries rather than matching words. To see what that adds over keyword matching, a set of operator questions was run through it, some worded so their surface terms point to one category while their meaning belongs to another. The questions were posed in Italian, the system's working language, and are translated here. Table 5.2 shows four representative cases.

Two operational questions carry misleading vocabulary (a "work order" and an "anomaly", a "bearing problem") that a keyword rule would read as technical or diagnostic. The classifier sends both to the operational pool, guided by what they actually ask (what the previous shift left, whether a colleague already dealt with it), and retrieves shift briefings and operator feedback. A third question, stating a live residual and health index, is routed correctly to the diagnostic configuration. In each case the meaning, not the surface form, decides the route: a lexical classifier could not do this.

The last question shows the limit and its safeguard together. A technical question about fretting corrosion falls just short of the classifier's margin guard and is routed to the diagnostic configuration instead. This is the conservative fallback of Section 4.4.1: when the top two categories are too close to separate, the query is widened to the mixed pool, not

narrowed to a single one. The cost is small: the question still retrieves its three technical chunks (the gearbox FMEA), so the evidence to answer it is present even when the label is not sharp.

The pattern matches the classifier's design. Operational queries, whose vocabulary is distinct, are separated cleanly; technical and diagnostic queries, close in meaning, are often merged into the diagnostic configuration, which still draws on the technical pool, so no relevant evidence is lost. The routing thus delivers its real value where it counts, keeping operator memory apart from the technical corpus, and degrades safely where the two genuinely overlap. Sharper separation of technical from diagnostic is left to Chapter 6.

***Table 5.2**. Classification of four operator queries by the semantic router.*

Query (translated from Italian)	Misleading surface cue	Similarity (tec · op · diag)	Routed to (retrieved stratum)
Should I open a work order, or was the anomaly already closed on the previous shift?	"work order", "anomaly" (diagnostic)	0.36 · 0.65 · 0.47	operational pool, 4 shift briefings
Did the colleague on the previous shift already fix the bearing problem?	"bearing problem" (technical)	0.43 · 0.54 · 0.47	operational pool, feedback + briefings
Residual of +9 °C on Gear_Oil, HI 0.20: real fault or transient?	–	0.63 · 0.37 · 0.75	diagnostic / mixed, technical + 1 briefing
What is fretting corrosion in bearings?	–	0.42 · 0.25 · 0.38	diagnostic / mixed (fallback), gearbox FMEA + 1 feedback

5.4.2 End-to-end diagnostic traces

The RAGAS evaluation of Section 5.3 measures the diagnostic layer in aggregate, on curated question–reference pairs; it does not show a diagnosis being assembled from a live alarm. This section gives that complementary view. Four alarms were drawn from turbine T06's 2017 blind stream (real records, not constructed cases), and each was submitted to the full pipeline exactly as the running system processes it: the most anomalous component, ranked on the normalised residual, sets a component-aware retrieval query; the query is embedded and matched against the two retrieval pools; and a constrained generation step returns a structured diagnosis together with a deterministic urgency. For every case the complete trace was recorded: the health index and residuals that raised the alarm, the retrieval query, the chunks returned with their sources and scores, the generated diagnosis, and the two urgency values. Selection used only quantities the detection layer itself computes, so no diagnostic outcome was pre-judged. The four cases are summarised in Table 5.3 and then read along two axes: whether the retrieved evidence discriminates between components, and how the

deterministic and probabilistic urgency signals diverge with severity. The diagnoses and the knowledge base are written in Italian, the working language of the modelled operation; quotations below give the original followed by an English translation.

***Table 5.3**. The four traced diagnostic cases (turbine T06, 2017 blind stream).*

Case	Timestamp	HI	Dominant component (residual)	Diagnosis	Urgency (det. / model)	Technical chunks retrieved	Operational chunk (score)
1	2017-11-10 13:40	0.217	Gearbox oil (+10.8 °C)	Developing gearbox lubrication fault	24 h / IMMEDIATA	DOC2#3, DOC3#10, DOC2#8	prior briefing (0.68)
2	2017-03-03 21:50	0.362	Generator bearing (+31.2 °C, isolated)	Generator-bearing fault	7 days / IMMEDIATA	DOC2#3, DOC2#8, DOC2#15	prior briefing (0.65)
3a	2017-12-12 13:10	0.078	Gearbox oil (+11.5 °C)	Gearbox lubrication fault	immediate / IMMEDIATA	DOC2#3, DOC3#10, DOC2#8	prior briefing (0.68)
3b	2017-12-12 18:00	0.392	Gearbox oil (+5.6 °C)	Gearbox lubrication fault	7 days / IMMEDIATA	DOC2#3, DOC3#10, DOC2#8	prior briefing (0.68)

Cases 1 and 2 test whether the retrieved evidence, and the diagnosis built on it, separate the gearbox from the generator rather than defaulting to the failure mode best represented in the knowledge base. In Case 1 (10 November 2017; HI = 0.217) the dominant residual is on the gearbox oil temperature, +10.8 °C, with smaller positive residuals on the gearbox bearing (+5.4 °C) and generator bearing (+8.5 °C). In Case 2 (3 March 2017; HI = 0.362) the pattern is inverted and sharper: the generator-bearing residual reaches +31.2 °C in isolation, while both gearbox channels stay near +3.5 °C. Component-aware query construction reflects this: Case 1 opens its query with “gearbox oil fault”, Case 2 with “generator bearing fault”, so the two alarms interrogate the knowledge base with different leading terms even though the rest of the query template is identical. All four alarms are classified as diagnosis-type queries and routed to both retrieval pools, each returning three technical chunks and one operational chunk.

Both retrievals return chunk DOC2#8, which states that a *“residuo anomalo ISOLATO su Gen_Bear_Temp_Avg … DISTINGUE il guasto generatore dal gearbox”* (“an anomalous residual isolated on Gen_Bear_Temp_Avg … distinguishes the generator fault from the gearbox”). The two traces apply the same chunk in opposite directions: in Case 1, where the anomaly is not isolated on the generator, the diagnosis is a developing gearbox lubrication fault; in Case 2, where it is, the diagnosis is a generator-bearing fault, its root-cause field

naming the isolated anomaly on Gen_Bear_Temp_Avg. The evidence that licenses the distinction is present in the retrieved record, and the generated text is consistent with it.

Two qualifications attach to the same traces. First, the top-ranked technical chunk in all four cases is DOC2#3, the account of T06's certified October-2017 gearbox progression, returned at a normalised score of 1.0; because scores are normalised to the leading chunk, that value should be read as "closest available" rather than an absolute match, and its recurrence reflects a small, T06-centric technical corpus in which the gearbox-failure narrative dominates. It does not override Case 2, whose generator diagnosis rests on DOC2#8, but it does show the retrieval favouring the best-documented event. Second, Case 2 also returns DOC2#15, a block of NBM performance metrics ($R^2 = 0.985$) largely unrelated to the fault, low-relevance retrieval consistent with the modest context-relevance score of Section 5.3.

Cases 3a and 3b isolate the urgency mechanism by holding everything else fixed. Both are gearbox-oil alarms on turbine T06 from the same day, 12 December 2017, about five hours apart; both select the same dominant component and retrieve the same four chunks. They differ only in severity: Case 3a is a deep alarm (HI = 0.078, +11.5 °C oil residual), Case 3b sits at the alarm boundary (HI = 0.392, +5.6 °C residual). Case 3a also shows why the residual is normalised before the dominant component is chosen: its raw generator-bearing residual, +12.2 °C, exceeds the oil residual in absolute terms, yet against each channel's own calibrated scale the oil anomaly is the larger, and it is the one selected.

The two urgency signals diverge across this gap. The deterministic rule, which maps the health index onto four bands, separates the cases: Case 3a is rated immediate, Case 3b seven days. The language model's urgency does not separate them: it returns the same maximal class, IMMEDIATA, for both, as it does for all four cases in Table 5.3. Its free-text recommendation is, moreover, identical in the two cases: *"Ispezione prioritaria del sistema di lubrificazione del gearbox e verifica del sistema di raffreddamento, da effettuare senza indugi"* ("priority inspection of the gearbox lubrication system and verification of the cooling system, to be carried out without delay"), even though the health index is five times lower in Case 3a than in Case 3b. Left to the generative layer alone, an operator facing a borderline alarm would be told to act "without delay" in the same terms as a near-critical one.

This is the concrete motivation for the division of responsibility introduced in Chapter 3: the probabilistic layer reasons and phrases, but the operational commitment, how soon to act, is

fixed by a deterministic rule over the health index and written to the audit trail as the authoritative value, with the model's own choice kept only for comparison. The trace also exposes a residual tension: in Case 3b the deterministic urgency is seven days, yet the retained free-text still reads *"senza indugi"* ("without delay"). Precedence resolves the contradiction (the deterministic field governs and the model's phrasing is subordinate), but the phrasing itself shows the generative layer's tendency to compress the severity scale, a point taken up in Section 5.7. The diagnosis nonetheless stays anchored to the retrieved text: Case 3b's root cause reproduces DOC3's wording that the anomaly is *"non spiegabile solo da variazioni ambientali"* ("not explicable by ambient variation alone"), and the residual bands it relies on are the data-driven NBM interpretation scale, not IEC limits, and are treated as such.

One feature common to all four traces points past the single diagnosis. Every retrieval returns, alongside its technical chunks, one operational chunk (a prior shift briefing, at scores of 0.65–0.68), so each diagnosis is formed with the previous shifts' open items within reach rather than from the sensor record alone. And every diagnosis, once generated, is written to the persistent audit trail with its evidence, its two urgency values, and a review field initialised to PENDING, the single controlled point at which human judgment enters the record. The four cases of this section stop at that PENDING state; Section 5.5 takes it forward, following the operator's review and the way an unresolved alarm is carried from one shift to the next.

5.5 Cross-shift knowledge continuity

Section 5.4 followed four diagnoses up to a single point: once each was recorded in the audit trail, its review field stood at PENDING. This section carries the account forward, into the hours and shift changes that follow. In doing so it demonstrates the third contribution of the thesis, the continuity of operational knowledge between shifts. The evidence is the system's own recorded operation on turbine T06 over 8–9 October 2017: an earlier run left, in the audit trail and the briefing store, a real sequence of alarms, operator reviews, and handovers, and that record is what is read here.

The open-anomaly block at the head of each briefing is not written by the language model. It is computed from the audit trail by a fixed rule. For each turbine–component pair, the system takes the alarms up to the shift boundary, discards everything up to and including the

operator's most recent correction, and reports what remains, labelled CONFERMATO if the operator has confirmed any alarm in it, PENDING otherwise. Such a case then persists from one briefing to the next, carrying its latest health index and urgency, until a correction closes it; a new alarm after a correction opens a fresh one.

The recorded sequence shows this directly. In the first shift the operator confirms a gearbox-oil anomaly on T06 (8 October, HI = 0.316, 48–72 h). The next two shifts register no new alarm, yet each briefing still opens with it, the handover noting that *"restano le anomalie aperte … dai turni precedenti"* ("the open anomalies from previous shifts remain"). A confirmed anomaly is thus carried, unchanged, through silent shifts and stated plainly in the handover. By the fourth recorded shift (9 October) it is still open, its health index updated to the latest reading, 0.277, now joined by a generator-bearing anomaly that alarmed in the interval (HI = 0.321, PENDING). The open list is not a snapshot of the current shift but a running record that each briefing inherits from the one before.

Not every alarm survives review, and the same record shows an operator closing one. Earlier in the first shift a generator-bearing alarm had been raised (8 October, HI = 0.304). The operator reviewed it, marked it CORRETTO, and recorded a cause and a note: *"Transitorio termico da rampa di carico, non degrado del cuscinetto"* ("a thermal transient from a load ramp, not bearing degradation"), with the added remark *"Verificato sui trend, falso allarme da transitorio"* ("checked against the trends, a false alarm from a transient"). The correction closes the case: it is absent from every open-anomaly list that follows. Yet closing an episode does not silence the component. When the generator alarms again on 9 October, a new episode opens, labelled PENDING, and takes its place in the handover. The operator's judgment settles the past without suppressing the future.

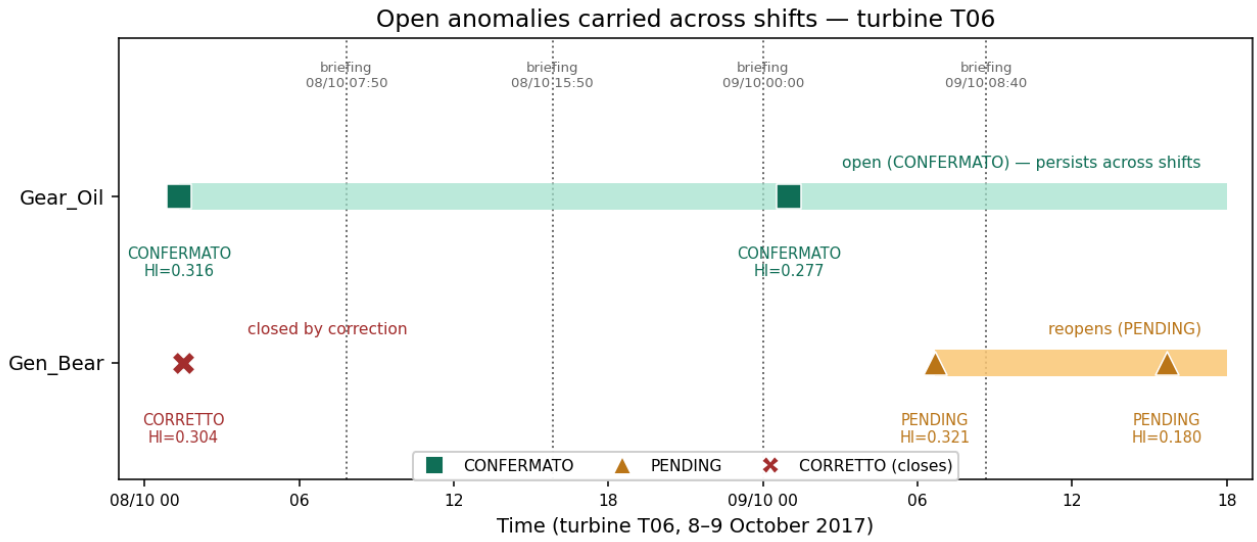

***Figure 5.3**. Open anomalies carried across shifts on turbine T06, 8–9 October 2017.*

That judgment is not confined to the audit field. The correction is also written as a feedback entry and indexed into the operational retrieval pool, so a later query can recover it. This closes, from the other side, the loop seen in Section 5.4, where every diagnosis retrieved one operational chunk: those chunks were briefings produced in an earlier session, which means the operational memory had already survived a restart and been rebuilt from disk. The same holds here: a shift's correction, note and briefing are written to human-readable files, re-ingested at start-up, and offered to the next shift's retrieval. The authoritative state is the audit trail, the briefing store, and the feedback entries; the embedding caches are derived from them and recomputable, never the source of truth.

5.6 Discussion

5.6.1 Traceability and epistemic separation

The question behind this thesis is not whether the model is right more often, but whether an operator can see why a diagnosis says what it says. That is what traceability means here, and it is the central claim the results support. The explainability literature of Section 2.5 frames the same shift: for closed models such as gpt-4o-mini, internal interpretability is out of reach, so trust has to be built at the level of the output. The traces of Section 5.4 show this at work. Each diagnosis lays out the chunks it drew on, their sources, and a set of structured fields that can be checked one by one; the urgency, in particular, is computed outside the model,

by the deterministic rule over the health index. The operator does not have to take the model's word: the evidence is on the record.

The clearest demonstration is the pair of component cases. Faced with a gearbox-dominant alarm and a generator-dominant one, the system retrieved the same distinguishing passage and applied it in opposite directions, and the diagnosis followed that evidence rather than the failure mode most common in the corpus. Traceability, then, is not a label pinned on after the fact; it is visible in the chain from residual to query to cited chunk to claim.

This works only because of how the knowledge base is built. Its stratification into two retrieval pools, a curated technical pool and a dynamic operational one, is the substrate that makes source attribution possible; here Gap 1 supports Gap 2 directly. A diagnosis can name where a claim comes from, and keep technical knowledge apart from operator memory, because the pools are separate by construction rather than by instruction to the model. This is the epistemic separation set out as a design principle in Chapter 3: enforced structurally, not left to the prompt.

The content of the technical pool matters for credibility, though not as a contribution in itself. The pool is filled with certified standards and peer-reviewed evidence rather than fabricated text, which makes the grounding believable in the wind-turbine setting where the system is tested. The value lies in the stratification and the output-level attribution, not in conformance to any standard; and the link from a residual to a failure mode stays a proposed, physically motivated heuristic, not a certified correspondence. Seen this way, the standards make the testbed more complete without becoming the point of the work.

One boundary should be stated plainly, and it returns in Section 5.7: the attribution reaches the retrieved document, not always the primary source quoted inside it. Within that boundary, traceability here is a property of the architecture rather than of a single output, the ground on which the thesis builds its case for trust.

5.6.2 Cross-shift continuity as a testbed of traceability

Traceability at a single moment is worth little if the diagnosis it grounds is lost at the next shift change. This is the point Section 2.7 draws from the handover literature: in the accidents it examines (Piper Alpha, Buncefield, Texas City), the knowledge existed, but it failed to cross from one shift to the next, and the failure was one of application rather than of understanding. Continuity is therefore where traceability is really tested, because it is where grounded knowledge has to survive a change of hands.

The results of Section 5.5 show the mechanism meeting that test on real records. A confirmed anomaly was carried forward through successive briefings, silent ones included, and stayed visible until an operator closed it; a corrected alarm dropped out and reopened only when the component alarmed again. The carry-forward is computed deterministically from the audit trail, not narrated by the model, so a slowly maturing fault cannot quietly fall out of the handover once the shift that first saw it has ended. Where the handover literature diagnoses a gap in application, the system answers with one in architecture: the good practice is enforced by construction.

The same continuity that carries an open anomaly forward also turns tacit knowledge into an explicit form. An operator's note, once written into the operational memory (Section 5.5), becomes retrievable by later diagnoses, so the informal reasoning that usually stays in an operator's head, or in a paper logbook, enters the system's searchable memory. This is the least-explored of the thesis's contributions, and the one that touches the dark-data problem of maintenance most directly.

Read together with Section 5.6.1, continuity extends traceability from a property of one output into a property of the system over time. A single diagnosis can be traced to its sources; a maturing anomaly can be traced across the shifts that observe it; and an operator's correction can be traced into the memory that later diagnoses draw on. The boundaries of this evidence (a single recorded sequence, and the bookkeeping limits of the open set) are taken up in Section 5.7.

5.6.3 Evaluating the system honestly

Reporting the RAGAS figures without qualifying them would misrepresent them. The same model, gpt-4o-mini, generated the diagnoses and then judged their faithfulness and relevance. Self-evaluation of this kind carries a known optimism, the self-preference bias documented by Wataoka et al. [33], so a high faithfulness score partly reflects a model marking its own work. The figures of Section 5.3 are read here, accordingly, as comparative indicators useful for ranking configurations of the system, not as absolute measures of its quality.

The low context-relevance score calls for the same care, in the opposite direction. It reads easily as a retrieval failure; it is better understood as a property of the metric. RAGAS scores context relevance sentence by sentence [32], whereas the system retrieves whole chunks, and the framework's own authors record this dimension as the hardest of the three to

estimate. The low value reflects the granularity of the measurement more than the quality of what was retrieved, a reading the uniform spread across all twenty-one questions supports, rather than one invented to excuse the number.

Because an automated metric that shares the model under test cannot settle the matter on its own, the chapter does not rest on it. The traces of Sections 5.4 and 5.5 give a second, independent check: each diagnosis is weighed against the evidence it cites and, where relevant, against the certified failure it concerns, without passing through the model-as-judge. This is the reasoning behind Gap 4's commitment to anchor the diagnostic claim to the IEC failure-mode taxonomy, a reference that does not shift with the model's own preferences. The two validations answer different questions: RAGAS asks, cheaply and at scale, whether the answers are generally faithful; the traces ask, case by case, whether a particular diagnosis is grounded and correct.

Neither is sufficient alone, and that is the point. A quantitative score carrying a self-preference bias, and a set of qualitative traces without statistical weight, are candid about different weaknesses; together they give a fuller account than either could alone. Treating evaluation this way, declaring its bias rather than hiding behind a single number, is itself part of what the thesis takes a trustworthy diagnostic system to require.

5.7 Real-world grounding, validity and limitations

The evidence of this chapter draws its weight from the data it runs on. Detection, diagnosis, and continuity were all exercised on the EDP wind farm dataset, real operational SCADA from four turbines across 2016 and 2017, and the diagnostic chain was checked against a documented failure, the T06 gearbox-bearing event of 17 October 2017. The system trained and ran blind to the failure log; the log served only as an after-the-fact reference. This grounding in real data with a certified outcome (Gap 3) is what separates the results from a demonstration on synthetic or idealised cases.

A boundary of a different kind concerns the detection layer, and the thesis draws it deliberately rather than stumbles on it. The Normal Behaviour Model is a standard thermal indicator built on three SCADA temperatures; detecting or predicting faults is not a contribution of this work, and the system should not be read as a predictive-maintenance detector. Section 5.2 shows why this matters on the T06 case: the certified failure, whose signature is primarily vibrational, leaves only a faint thermal trace, and the aggregate Health

Index is disturbed more by winter conditions than by the failure itself. A detector able to catch such a fault would need vibration and further channels a thermal-SCADA system does not observe, as the condition-monitoring literature of Chapter 1 makes clear. The T06 event therefore serves here as a real reference against which the diagnostic traces can be read, not as evidence of early detection.

The grounding also sets the outer limit of what the results establish. The validation rests on one turbine of one open dataset, on a small set of real but deliberately chosen alarms, and on a single recorded sequence of shifts. It shows the mechanisms work as intended and exposes their limits, but it does not establish their behaviour statistically or across turbines, sites, and sensor sets. The urgency thresholds, in particular, were calibrated on this turbine's audit history and are presented as data-driven values open to recalibration elsewhere, not as universal cut-offs.

Two limits surface in the diagnostic traces. The attribution reaches the retrieved document rather than the primary source quoted inside it, the diagnosis names DOC2, not the España or Carroll study that document draws on, so traceability, though real, stops one level short of the original reference. And the retrieval leans toward the best-documented event: the account of the certified T06 failure is the top-ranked technical chunk in every trace, and one case also returns a block of model-performance figures of little relevance to the fault. Neither misleads the diagnosis, but both reflect a small, T06-centred corpus that a larger, more balanced knowledge base would even out.

The generative step has two tendencies of its own. It compresses the severity scale, free-text urgency and recommended action barely change between a deep alarm and a borderline one, which is why the deterministic rule, not the model, sets the urgency written to the record. And it can present a data-driven residual threshold as though it were a certified limit; the knowledge base keeps the two apart, but the generation can blur them, a point to watch rather than one the present system fully resolves.

The continuity mechanism carries two simplifications, both noted in Section 5.6.2. Open anomalies are tracked one component at a time, so two distinct faults on the same channel would be folded into a single case; and the operational memory only grows, leaving an orphan feedback entry behind if a correction is later revised. Neither weakens the continuity the chapter demonstrates, but both mark where its bookkeeping is approximate.

Two points of validity, finally, bound the reading of the numbers themselves. The RAGAS metrics measure faithfulness and relevance, which stand in for traceability rather than measuring it directly, a construct limit inherent to any automated proxy. And, as Section 5.6.3 sets out, generator and judge are the same model, so the absolute scores lean optimistic and are read here only as comparative indicators. Taken together, these limits do not overturn the chapter's findings; they fix the boundary of what has been shown, and mark the ground the concluding chapter builds on.

6. Conclusions

6.1 Summary

The aim of this thesis was to make the output of an LLM-based diagnostic system trustworthy in a safety-critical industrial setting. Not by making the model right more often, but by making each diagnosis traceable, grounded in an auditable knowledge base, and open to operator correction over time. Chapter 1 framed the problem around three limitations of current LLM-assisted maintenance: a knowledge base treated as a single undifferentiated corpus, a diagnostic output whose provenance cannot be recovered, and diagnostic knowledge lost at the change of shift.

The response was AURA-PHM. A Normal Behaviour Model flags thermal anomalies on wind-turbine SCADA data. A retrieval-augmented reasoning layer turns an alarm into a structured, source-attributed diagnosis. A human-oversight layer records the operator's judgement and carries open items across shifts. The framework was set out in general terms in Chapter 3, instantiated on the EDP wind-farm dataset in Chapter 4, and validated in Chapter 5 on turbine T06's 2017 blind stream, quantitatively with RAGAS and qualitatively through real classification, diagnostic, and shift-continuity traces.

The three research objectives were met, each within the limits the thesis states openly. The diagnostic layer produces outputs that can be traced back to the chunks and sources that informed them (Objective 1). The knowledge base is stratified into two retrieval pools of different epistemic status, so a citation can be placed as technical or operational (Objective 2). Within that structure, a residual is linked to a qualitative failure-mode taxonomy, a mapping proposed on physical grounds, not a certified correspondence. And a note entered by an operator in one shift is captured, indexed, and made retrievable in the next (Objective 3). What the thesis does not claim is a contribution to fault detection: the Normal Behaviour Model is a standard thermal indicator, and the value of the system begins where the alarm ends.

6.2 Contributions

The contribution of this thesis is not a single new technique. It is the integration of five elements into one working framework, validated on real data. Each is modest on its own. Together they make the diagnosis traceable, grounded, and durable across shifts.

The first is the framework itself: a complete, end-to-end prototype in which a Normal Behaviour Model, a retrieval-augmented reasoning layer, and a human-oversight layer run as one pipeline, on the open EDP dataset and against a documented failure. The second is the stratified knowledge base: two retrieval pools of different epistemic status, separated by construction. Its value is structural, not tied to the material that fills it. Here the technical pool holds certified standards; in a production system the same design would hold proprietary documentation, with the same auditability.

The third is the deterministic rule that turns the health index into an urgency level, computed outside the model. It corrects a known weakness of LLM classification under a cautious label set, and keeps the model's own urgency alongside the rule's, for comparison. The fourth is the operator-feedback mechanism: a structured confirmation or correction, a free-text note, and three channels of persistence that make an operator's judgement retrievable in later shifts (a concrete instance of turning tacit knowledge into explicit form). The fifth is the evaluation itself, quantitative with RAGAS and qualitative through end-to-end traces, reported with its own biases stated rather than hidden.

Two boundaries keep these claims honest. Fault detection is not one of the contributions: the Normal Behaviour Model is a standard thermal indicator, and the system is not offered as a predictive detector. And the residual-to-failure-mode link is a proposed mapping, grounded in physics and in the T06 evidence, not a certified correspondence. Within those boundaries, the framework does what it set out to do: it makes an LLM diagnosis one can follow, question, and carry forward.

6.3 Limitations and future directions

Chapter 5 examined the limitations of the results in detail. Here they are gathered, set beside the implementation limits of the prototype, and each pointed toward the work that would address it.

The most important boundary is one of scope. The system does not detect faults, and is not meant to. Its Normal Behaviour Model is a standard thermal indicator, and three SCADA temperatures capture a bearing fault whose signature is vibrational only faintly, as the T06 case made plain. A detector for such faults would need vibration and further channels; extending the sensing beyond temperature is what would let the detector, not only the diagnosis, carry weight.

The validation is narrow in the same spirit. It rests on one turbine, one dataset, one recorded sequence of shifts: enough to show the mechanisms work, not to establish them statistically or across sites. Testing on more turbines and farms, in time on other domains, and re-calibrating the urgency thresholds would show how far the design travels.

A second group concerns how exact the evidence is. The attribution reaches the source document, but not always the primary reference quoted inside it; naming that original source is a clear refinement. The evaluation, too, shares its judge: the generator and the RAGAS assessor are the same model, so the scores lean optimistic. An independent judge, tested on a broader set of operational and diagnostic queries, would measure the system more fairly.

The retrieval that feeds the diagnosis is only partly tuned. The query classifier separates operational questions well but merges technical and diagnostic ones, whose centroids sit almost parallel. Here the path forward is unusually concrete: removing the shared domain component pulls the two apart in analysis (from 0.79 to −0.03 in cosine similarity), and a trained classifier would go further. The chunking is fixed too, split by character count with hand-set breadth and thresholds; semantic chunking and adaptive retrieval would sharpen both the context and the metric that scores it.

The last limits are those of a prototype rather than a product. Runs are not exactly reproducible: generation uses a non-zero temperature without a fixed seed, and both embedding and generation depend on a hosted model that can change. There is no automated test suite, and the reliance on a commercial API brings cost, latency, and privacy with it. Pinning the model, fixing the randomness, adding tests, and moving to a local model are the ordinary steps between a working prototype and a deployable one.

None of these undermines the contribution. Together they mark the distance still to travel.

6.4 Concluding remarks

The question this thesis began with was not whether a language model can diagnose a fault, but whether its diagnosis can be trusted where the stakes are high. The answer it offers is a modest one, and deliberately so. Trust, here, is not the model being right more often. It is the operator being able to see where a diagnosis comes from, to correct it, and to pass it on.

That shift, from accuracy to accountability, is what the framework is built around. A diagnosis that cites its sources can be checked; an urgency fixed by a rule rather than by the

model can be relied on; a correction written in one shift can inform the next. None of these depends on the model being infallible, which is why they hold even when it is not.

The work stops well short of a deployable system, and says so throughout: it is a prototype, validated on a single turbine, honest about a biased evaluation and about a detector that only triggers the reasoning it cannot replace. But within those limits it makes a case easy to lose in the rush toward larger models: that in a safety-critical setting, a diagnosis one can follow, question, and carry forward may matter more than one that is merely confident.

References

[1] I. U. Hassan, K. Panduru, and J. Walsh, “Predictive Maintenance in Industry 4.0: A Review of Data Processing Methods,” *Procedia Computer Science*, vol. 257, pp. 896–903, 2025. DOI: 10.1016/j.procs.2025.03.115.

[2] R. Sala, E. Francalanza, and S. Arena, “A review on three decades of manufacturing maintenance research: past, present and future directions,” *Production & Manufacturing Research*, vol. 13, no. 1, art. 2469037, 2025. DOI: 10.1080/21693277.2025.2469037.

[3] Y. W. Tan and J. Carroll, “LLMs in Wind Turbine Gearbox Failure Prediction,” *Energies*, vol. 18, no. 17, art. 4659, 2025. DOI: 10.3390/en18174659.

[4] J. Carroll, A. McDonald, and D. McMillan, “Failure rate, repair time and unscheduled O&M cost analysis of offshore wind turbines,” *Wind Energy*, vol. 19, no. 6, pp. 1107–1119, 2016. DOI: 10.1002/we.1887.

[5] J. Carroll, S. Koukoura, A. McDonald, A. Charalambous, S. Weiss, and S. McArthur, “Wind turbine gearbox failure and remaining useful life prediction using machine learning techniques,” *Wind Energy*, vol. 22, no. 3, pp. 360–375, 2019. DOI: 10.1002/we.2290.

[6] R. K. Pandit, D. Astolfi, and I. Durazo Cardenas, “A Review of Predictive Techniques Used to Support Decision Making for Maintenance Operations of Wind Turbines,” *Energies*, vol. 16, no. 4, art. 1654, 2023. DOI: 10.3390/en16041654.

[7] F. Castellani, F. Natili, D. Astolfi, and Y. Vidal, “Wind turbine gearbox condition monitoring through the sequential analysis of industrial SCADA and vibration data,” *Energy Reports*, vol. 12, pp. 750–761, 2024. DOI: 10.1016/j.egyr.2024.06.041.

[8] S. España, J. Sánchez, B. Puruncajas, F. Castellani, C. Tutivén, and Y. Vidal, “Early detection of gearbox faults in wind turbines using a fine-tuned transformer encoder,” *Results in Engineering*, vol. 30, art. 109915, 2026. DOI: 10.1016/j.rineng.2026.109915.

[9] S. Wang, K. Wang, P. Zeng, D. Li, J. Yang, S. Zhang, and L. Cao, “A Large Model for Integrated Diagnosis, Prognosis, and Decision-Making in Intelligent Operation and Maintenance,” *Proc. 7th Int. Conf. on System Reliability and Safety Engineering (SRSE)*, IEEE, 2025. DOI: 10.1109/SRSE67406.2025.11357424.

[10] A. Carvalho, F. Boyle, J. Walsh, and J. Daly, “Building Artificial Intelligence Agents with Large Language Models for Autonomous Interpretation of Equipment Sensor Data: P2,” , IEEE, 2025.

[11] H. A. A. M. Qaid, B. Zhang, S. Su, D. Li, S.-K. Ng, and W. Li, “Large language models for explainable fault diagnosis of machines,” *Engineering Applications of Artificial Intelligence*, vol. 163, art. 113131, 2026. DOI: 10.1016/j.engappai.2025.113131.

[12] C. Tsallis, P. Papageorgas, R. A. Munteanu, and S. Dellagi, “Large Language and Foundation Models for Machinery Health Monitoring: A Systematic Review,” *Applied Sciences*, vol. 16, no. 5, art. 2493, 2026. DOI: 10.3390/app16052493.

[13] H. Mirshekali et al., “A Review of Large Language Models for Energy Systems: Applications, Challenges, and Future Prospects,” *IEEE Access*, vol. 13, pp. 163149–163185, 2025.

[14] X. Fan, X. Wang, and Y. Liu, “Synergistic RAG: A Dual-Path Knowledge Retrieval Framework for Industrial Fault Diagnosis,” *Proc. 5th Int. Conf. on Electronic Information Engineering and Computer Communication (EIECC)*, IEEE, 2025, pp. 460–464.

[15] H. Wang, M. Qi, Z. Wang, and Q. Yang, “MS-RAG: A Multimodal Retrieval-Augmented Framework for Digital Twins with Knowledge Graph Reasoning,” *Proc. 10th Int. Conf. on Electronic Technology and Information Science (ICETIS)*, IEEE, 2025, pp. 307–311.

[16] Q. Nie, J. Geng, D. Tang, and C. Liu, “Industrial knowledge-enhanced fault diagnosis method: Integrating LLM and knowledge graph for fault reasoning and maintenance recommendation in CNC machine tools,” *Computers & Industrial Engineering*, vol. 214, art. 111879, 2026. DOI: 10.1016/j.cie.2026.111879.

[17] T. Morioka, T. Kono, and T. Nishida, “Automatic construction of asset knowledge graph with large language model,” *Procedia CIRP*, vol. 135, pp. 762–767, 2025. DOI: 10.1016/j.procir.2025.01.097.

[18] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, vol. 33, pp. 9459–9474, 2020.

[19] N. R. Amaliah, B. Tjahjono, and V. Palade, “Human-in-the-Loop XAI for Predictive Maintenance: A Systematic Review of Interactive Systems and Their Effectiveness in Maintenance Decision-Making,” *Electronics*, vol. 14, art. 3384, 2025. DOI: 10.3390/electronics14173384.

[20] D. Kim, Y. S. Sim, S. Yoon, S. Yoon, S. Yoa, S. Lee, and W. Lim, “AEGIS: Toward Expert-in-the-loop Industrial Anomaly Detection,” *Proc. 40th AAAI Conference on Artificial Intelligence (AAAI-26)*, 2026, pp. 41619–41620.

[21] J. Cárcel-Carrasco and J.-A. Cárcel-Carrasco, “Analysis for the Knowledge Management Application in Maintenance Engineering: Perception from Maintenance Technicians,” *Applied Sciences*, vol. 11, art. 703, 2021. DOI: 10.3390/app11020703.

[22] W.-C. Shen and F.-R. Lin, “The Design of AI-Enabled Experience-Based Knowledge Management System to Facilitate Knowing and Doing in Communities of Practice,” *Knowledge Management in Organisations (KMO 2024), CCIS vol. 2152*, Springer, 2024, pp. 292–303.

[23] B. Puruncajas, F. Castellani, Y. Vidal, and C. Tutivén, “Use of Artificial Neural Networks and SCADA Data for Early Detection of Wind Turbine Gearbox Failures,” *Machines*, vol. 13, art. 746, 2025. DOI: 10.3390/machines13080746.

[24] A. Verma, D. Zappalá, S. Sheng, and S. J. Watson, “Wind turbine gearbox fault prognosis using high-frequency SCADA data,” *Journal of Physics: Conference Series*, vol. 2265, art. 032067, 2022. DOI: 10.1088/1742-6596/2265/3/032067.

[25] K. Kijanowski, T. Barszcz, and P. B. Dao, “A Cluster-Based Filtering Approach to SCADA Data Preprocessing for Wind Turbine Condition Monitoring and Fault Detection,” *Energies*, vol. 18, art. 5954, 2025. DOI: 10.3390/en18225954.

[26] International Electrotechnical Commission, *IEC 61400-4:2025, Wind Energy Generation Systems, Part 4: Design Requirements for Wind Turbine Gearboxes*, 2nd ed., May 2025.

[27] International Organization for Standardization, *ISO 20816-21:2025, Mechanical Vibration, Measurement and Evaluation of Machine Vibration, Part 21: Horizontal Axis Wind Turbines*, 1st ed., May 2025.

[28] International Electrotechnical Commission, *IEC TS 61400-9:2025, Wind Energy Generation Systems, Part 9: Probabilistic Design Measures for Wind Turbines*, 1st ed., July 2025.

[29] K. Liu, S. Zhang, Y. Wang, H. Zhang, and W. Zhang, "RADAR: Agent-Guided Retrieval with Physics-Based Consistency Filtering for Aviation Fault Diagnosis," *Proc. 7th Int. Conf. on Frontier Technologies of Information and Computer (ICFTIC)*, IEEE, 2025, pp. 1152–1158. DOI: 10.1109/ICFTIC68075.2025.11324487.

[30] H. Luo and L. Specia, "From Understanding to Utilization: A Survey on Explainability for Large Language Models," *arXiv preprint* arXiv:2401.12874, 2024.

[31] M. N. Ahangar, Z. A. Farhat, and A. Sivanathan, "AI Trustworthiness in Manufacturing: Challenges, Toolkits, and the Path to Industry 5.0," *Sensors*, vol. 25, no. 14, art. 4357, 2025. DOI: 10.3390/s25144357.

[32] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "Ragas: Automated Evaluation of Retrieval Augmented Generation," *arXiv preprint* arXiv:2309.15217, 2023.

[33] K. Wataoka, T. Takahashi, and R. Ri, "Self-Preference Bias in LLM-as-a-Judge," *arXiv preprint* arXiv:2410.21819, 2025.

[34] J. Wilkinson and R. Lardner, "Pass It On! Revisiting Shift Handover After Buncefield," *Hazards XXIII*, Symposium Series No. 158, Institution of Chemical Engineers, 2012, pp. 113–118.

Appendices

These appendices reproduce the operative artefacts of the AURA-PHM prototype so that every claim in Chapter 5 can be traced to the exact material that produced it. The artefacts are reproduced verbatim in the system's working language, Italian, exactly as they exist in the implementation; each appendix opens with a note in English, and the key structural elements (the failure-mode taxonomy, the response schema, the audit-record fields) are given in English translation. The figures reported here are the recorded outputs behind the thesis body, not fresh executions: they therefore match Table 5.3, the RAGAS results of Section 5.3, and the shift-continuity record of Section 5.5. The full source code and the SCADA dataset are not reproduced here; the code is available in the project repository and the EDP open dataset from its public source.

Appendix A: The stratified knowledge base: source documents

The four curated documents that constitute the technical retrieval pool (Section 4.3.1). They are the source of truth against which every diagnosis is attributed: the chunk identifiers cited in Appendix D (for example DOC2#8, DOC2#3) index into these files. The residual-to-failure-mode association within DOC1 is marked in the document itself as a methodological proposal of this work, anchored to but distinct from the certified IEC 61400-4:2025 taxonomy.

A.1: DOC1 Gearbox FMEA and failure-mode taxonomy

Structure (English). DOC1 catalogues five failure modes, each anchored to the IEC 61400-4:2025 Table 10/11 classification:

```
1. Planetary bearing cage fracture (Planet 1 bearing). Severity: critical. IEC
category A2 (cage or ring fracture).
2. Abrasive wear of planetary gears (Planet 3 gear teeth). Severity: high. IEC
category B (abrasive wear).
3. Fretting corrosion of bearings (planetary 2/3, IMS, LSS, HSS). Severity:
moderate. IEC category B (fretting).
4. HSS pinion scuffing (High Speed Shaft pinion). Severity: high. IEC category B
(scuffing).
5. Generator bearing overheating (Gen_Bear_Temp_Avg). Severity: high. IEC category
B (overheating).
```

Verbatim source (Italian):

```
FMEA GEARBOX TURBINA EOLICA — Failure Mode and Effects Analysis
Documento tecnico per sistema AURA-PHM — Knowledge Base stratificata


Basato su: España et al. (2026) Results in Engineering 30, 109915
           Tan & Carroll (2025) Energies 18, 4659
           Castellani et al. (2024) Energy Reports 12, 750-761


Fonti normative certificate:
           BS EN IEC 61400-4:2025 - Design requirements for wind turbine gearboxes
           BS ISO 20816-21:2025 - Mechanical vibration, horizontal axis wind turbines


NOTA METODOLOGICA: le soglie sui residui sono derivate data-driven dal Normal
Behaviour Model (NBM) addestrato sui dati EDP 2016 (T06). La tassonomia dei
failure mode e' certificata IEC 61400-4:2025 Table 10/11. Il collegamento
residuo-failure mode e' una proposta metodologica di questo lavoro, fondata su
```

ragionamento fisico ed evidenza empirica (caso T06), ancorata alla tassonomia IEC.

```
================================================================
MODO DI GUASTO 1: FRATTURA GABBIA CUSCINETTO PLANETARIO
================================================================
Componente: Cuscinetti planetari stadio 1 (Planet 1 bearing)
Severità: CRITICA
Classificazione IEC 61400-4:2025 Table 10/11: Categoria A2 (cage or ring fracture)
  — failure mode deterministico, cumulativo, senza modelli di calcolo validati.
  Detection requirement (Table 11): ispezione visiva, no surface cracks.
Causa radice: Disallineamento eccessivo o forze vibranti anomale
  che sottopongono la gabbia a stress superiori al limite.
  In condizioni di sottocarico i rulli possono inclinarsi fino
  a 90 gradi, aumentando lo stress sulla gabbia fino a frattura.
  (i 90 gradi di inclinazione dei rulli sono documentati da España et al.
  2026 Section 3; il meccanismo generale di skidding dei rolling element e'
  descritto anche in IEC 61400-4:2025 Clause 6.2.1.4, senza il valore
  specifico dei 90 gradi)
Sintomi rilevabili da SCADA:
  - Gear_Oil_Temp_Avg: residuo positivo progressivo > 8 degC
  - Gear_Bear_Temp_Avg: residuo positivo > 5 degC
  - Pattern: crescita lenta nelle prime 2-4 settimane,
    poi accelerazione nelle ultime 7-14 giorni pre-guasto
  - Gen_Bear_Temp_Avg: tipicamente nella norma (guasto localizzato)
  - Potenza e RPM: normali fino alle fasi avanzate
Effetti: Rilascio di detriti metallici nel lubrificante.
  I detriti danneggiano progressivamente altri componenti
  (ingranaggi, cuscinetti IMS, LSS, HSS).
  Rischio di guasto a cascata sull'intero gearbox.
Tempo tipico di progressione da SCADA:
  Rilevabile con anticipo di 4-16 settimane prima del guasto
  confermato. (España et al. 2026, Section 6 - WT8 case study:
  rilevato 4 mesi prima del guasto certificato)
Azione raccomandata:
  HI < 0.60 e residuo Gear_Oil > 5 degC per > 48h:
    → Analisi lubrificante entro 48 ore
    → Pianificare ispezione visiva entro 7 giorni
  HI < 0.40 e residuo Gear_Oil > 10 degC per > 24h:
    → Analisi lubrificante urgente entro 24 ore
    → Ispezione tecnica entro 48-72 ore
    → Valutare riduzione carico operativo
  HI < 0.20:
    → Considerare fermo turbina per ispezione immediata


================================================================
MODO DI GUASTO 2: USURA ABRASIVA INGRANAGGI PLANETARI
================================================================
Componente: Planet 3 gear teeth, ingranaggi stadio planetario
Severità: ALTA
Classificazione IEC 61400-4:2025 Table 10/11: Categoria B (abrasive wear)
  — failure mode stocastico, alcuni modelli disponibili.
  Detection requirement (Table 11): ispezione visiva secondo ISO 15243.
Causa radice: Particelle dure nel lubrificante forzate sulle
  superfici dei denti degli ingranaggi, rimozione progressiva
  di materiale. Risulta da filtrazione inadeguata o detriti
  provenienti da guasti upstream (es. cage failure).
  (España et al. 2026, Section 3 - Abrasive Wear; requisiti di pulizia
  lubrificante in IEC 61400-4:2025 Clause 6.6.9, indentation da debris
  in Clause 6.2.1.10)
Sintomi rilevabili da SCADA:
  - Gear_Oil_Temp_Avg: residuo positivo persistente > 6 degC
  - Gear_Bear_Temp_Avg: residuo moderato > 3 degC
  - Pattern: crescita lenta e progressiva, tipicamente lineare
  - Rumore operativo aumentato (non rilevabile da SCADA standard)
  - Efficienza ridotta: potenza leggermente inferiore al atteso
    a parità di vento
Effetti: Compromissione dell'integrità del profilo del dente.
  Rischio crescente di mesh failure dell'ingranaggio.
  Accelerazione dell'usura con il tempo.
Tempo tipico di progressione: Settimane-mesi.
  SCADA può rilevare con anticipo di 2-8 settimane.
  (Castellani et al. 2024 - gearbox condition monitoring)
Azione raccomandata:
```

```
  HI < 0.60:
    → Aumentare frequenza analisi olio (ogni 2 settimane)
    → Verificare sistema di filtrazione
  HI < 0.40:
    → Analisi olio urgente - cercare particelle metalliche
    → Ispezione endoscopica gearbox entro 72 ore
    → Verificare integrità del sistema di lubrificazione


================================================================
MODO DI GUASTO 3: CORROSIONE PER FRETTING CUSCINETTI
================================================================
Componente: Cuscinetti planetari 2 e 3, cuscinetti IMS, LSS, HSS
Severità: MODERATA
Classificazione IEC 61400-4:2025 Table 10/11: Categoria B (fretting)
  – failure mode stocastico. IEC 61400-4:2025 Clause 6.2.1.6 descrive la
  fretting corrosion come wear process da micro-movimenti relativi sotto
  carico, esacerbato dalla presenza di umidità.
Causa radice: Micro-movimento relativo tra parti a stretto
  contatto (sedi cuscinetti, sedi ingranaggi). Produce
  cricche superficiali e pattern di ossidazione che agiscono
  come concentratori di stress.
  (España et al. 2026, Section 3 - Fretting Corrosion)
  Trovato in: Planet 2 e 3 bearings, IMS bearing (R-end e NR-end),
  LSS bearing (NR-end), Planet carrier bearing, HSS bearing.
Sintomi rilevabili da SCADA:
  - Gear_Bear_Temp_Avg: residuo positivo moderato 3-8 degC
  - Gear_Oil_Temp_Avg: residuo positivo lieve 2-5 degC
  - Pattern: fluttuazioni nel residuo, non necessariamente
    monotone - caratteristica distintiva dalla usura abrasiva
  - Variabilità aumentata nei residui (std residuo in crescita)
Effetti: Indebolimento progressivo strutturale.
  Riduzione vita utile del componente.
  Rischio frattura sotto carico elevato.
Azione raccomandata:
  HI < 0.60:
    → Monitoraggio ravvicinato - campionamento ogni 24h
    → Verifica condizioni operative (evitare carichi impulsivi)
  HI < 0.40:
    → Ispezione tecnica gearbox entro 5 giorni
    → Analisi vibrazionale se disponibile


================================================================
MODO DI GUASTO 4: SCORING PIGNONE HSS (ALTA VELOCITA')
================================================================
Componente: HSS pinion (High Speed Shaft pinion)
Severità: ALTA
Classificazione IEC 61400-4:2025 Table 10/11: Categoria B (scuffing)
  – failure mode stocastico, NON-cumulativo (può manifestarsi
  improvvisamente). Detection requirement (Table 11): ispezione visiva
  secondo ISO 14635-1:2023. IEC 61400-4:2025 Clause 6.1.2.3 richiede che la
  capacità di scuffing sia calcolata alla massima temperatura operativa
  del gearbox.
Causa radice: Rottura del film di olio, contatto metallo-metallo
  a temperature elevate. Porta a strappo e saldatura del materiale
  superficiale. Causato da eventi ad alto carico.
  (España et al. 2026, Section 3 - Scoring)
Sintomi rilevabili da SCADA:
  - Gear_Oil_Temp_Avg: picchi acuti di residuo > 15 degC
    seguiti da rientro parziale - pattern a dente di sega
  - Pattern: eventi acuti non necessariamente progressivi
  - Correlazione con picchi di potenza/carico elevati
  - Gear_Bear_Temp_Avg: può rimanere moderato
Effetti: Danni alla superficie del pignone.
  Riduzione efficienza trasmissione.
  Rischio guasto improvviso ad alta velocità.
Azione raccomandata:
  Qualsiasi picco residuo Gear_Oil > 15 degC:
    → Analisi lubrificante immediata
    → Ridurre carico operativo se possibile
    → Ispezione entro 48 ore


================================================================
MODO DI GUASTO 5: SURRISCALDAMENTO CUSCINETTI GENERATORE
```

```
================================================================
Componente: Generator bearings (Gen_Bear_Temp_Avg)
Severità: ALTA
Classificazione IEC 61400-4:2025 Table 11: Categoria B (overheating,
  detection visiva no discoloration da calore secondo ISO 7146-1).
  Per rolling bearing del generatore vedere anche rolling contact
  fatigue Categoria A1.
Causa radice: Usura cuscinetti generatore, problemi di
  lubrificazione generatore, sovraccarico elettrico.
  Indipendente dal gearbox.
  (Tan & Carroll 2025 - bearing overheating failure mode)
Sintomi rilevabili da SCADA:
  - Gen_Bear_Temp_Avg: residuo positivo > 8 degC ISOLATO
  - Gear_Oil_Temp_Avg e Gear_Bear_Temp_Avg: tipicamente normali
  - Questo pattern DISTINGUE il guasto generatore dal guasto gearbox
  - Pattern: può essere progressivo o acuto
Effetti: Danni ai cuscinetti del generatore.
  Rischio fermo generatore non pianificato.
  Downtime elevato per sostituzione cuscinetti generatore.
Azione raccomandata:
  Gen_Bear residuo > 8 degC con Gear_Oil nella norma:
    → Problema localizzato al generatore, non al gearbox
    → Ispezione generatore entro 48-72 ore
    → Verificare sistema di raffreddamento generatore
    → Non necessariamente richiede fermo gearbox
Canale diagnostico complementare (non implementato in questo sistema):
  ISO 20816-21:2025 Table A.1 definisce zone di valutazione vibrazione per
  generatori con rolling bearing (velocity Zone B/C 6,0 mm/s, C/D 10 mm/s;
  acceleration B/C 10 m/s2, C/D 16 m/s2). Sono soglie su canale vibrazionale
  distinto dal residuo termico SCADA, citate come riferimento normativo per
  un eventuale secondo canale di conferma, non applicabili ai residui termici.


================================================================
LIMITI ASSOLUTI DI SICUREZZA CERTIFICATI (IEC 61400-4:2025)
================================================================
Limiti assoluti di shutdown definiti dallo standard, distinti dalle soglie
di residuo NBM (che sono soglie di anticipo, su valori molto inferiori):
  - Temperatura bulk lubricant: shutdown a 85 degC (media 10 min)
    [IEC 61400-4:2025 Clause 6.6.7.2]
  - Temperatura rolling bearing: shutdown a 95 degC (media 1 min al ring
    stazionario), limita il contatto a 105 degC [Clause 6.6.7.3.2]
  - Temperatura plain bearing: shutdown a 115 degC, limita il contatto
    a 125 degC [Clause 6.6.7.3.3]
Valore aggiunto del NBM: i residui rilevano il degrado MOLTO PRIMA che le
temperature assolute raggiungano questi limiti. Un residuo di 8 degC non
significa olio a 85 degC, ma 8 degC piu' caldo del previsto per quel regime,
anticipando il problema mentre la temperatura assoluta e' ancora nella norma.


================================================================
MONITORAGGIO LUBRIFICANTE PER CONFERMA (IEC 61400-4:2025 Clause 6.6.8)
================================================================
Quando un residuo termico attiva il sospetto di un failure mode legato alla
lubrificazione (modi 2 e 4), la conferma richiede analisi del lubrificante.
IEC 61400-4:2025 Clause 6.6.8 specifica il monitoraggio obbligatorio di:
  - viscosità a 40 degC e 100 degC (ISO 3104)
  - numero acido (ISO 6618) o neutralization number (ISO 6619)
  - elementi metallici da usura (ASTM D5185)
  - pulizia / particle count (ISO 4406)
  - contenuto d'acqua (ASTM D6304)


================================================================
NOTA METODOLOGICA FONDAMENTALE
================================================================
I residui SCADA di temperatura non possono identificare
univocamente un modo di guasto specifico.
(España et al. 2026, Section 6.1: "SCADA temperature residuals
cannot uniquely identify a specific failure mode")
(Castellani et al. 2024: SCADA individua la turbina sospetta
e il periodo; la localizzazione precisa richiede CMS/vibrazioni)
Il sistema AURA-PHM usa i residui per:
  1. Rilevare la presenza di anomalia termica (L2 - NBM)
  2. Caratterizzare il pattern (componente, direzione, persistenza)
  3. Identificare il modo di guasto PIU' PROBABILE per il pattern
```

```
     osservato (L3 - RAG-LLM)
  4. Raccomandare l'azione appropriata con urgenza temporale
La diagnosi finale richiede sempre conferma da parte
dell'operatore tecnico qualificato (Human-in-the-Loop).
Questo approccio e' coerente con il framework di critical system analysis
(FMEA/FMECA) richiesto da IEC 61400-4:2025 Clause 4.5 e con il processo di
design verification and validation di Clause 7.
```

A.2: DOC2 EDP fault archive (empirical stratum)

```
FAULT ARCHIVE EMPIRICO — Wind Farm EDP
Casi storici documentati con pattern di residuo NBM
Fonte: EDP Open Data Failure Log 2017 + analisi NBM AURA-PHM
Dataset: Wind-Turbine-SCADA-signals-2016/2017 (EDP Open Data)


Fonti empiriche aggiuntive:
  España et al. (2026) Results in Engineering 30, 109915 (ispezione fisica caso WT8/T06)
  Carroll et al. (2015) Wind Energy - dati di affidabilità di campo (~350 turbine offshore)
═══════════════════════════════════════════════════════════════
CASO STORICO 1: T06 — GEARBOX BEARINGS DAMAGED
═══════════════════════════════════════════════════════════════
Turbina: T06
Data guasto certificata: 17 Ottobre 2017
Tipo guasto: Gearbox bearings damaged
Fonte: EDP Open Data - opendata-wind-failures-2017
Comportamento residui NBM nella finestra pre-guasto (modello addestrato su 2016, valutazione cieca
2017):
Nella finestra 3-16 ottobre 2017 i residui termici del gearbox mostrano una lieve elevazione
rispetto alla loro baseline annuale, non una degradazione progressiva netta:
  Gear_Oil_Temp residuo medio: +0.45 degC (baseline annua +0.25; delta +0.20)
  Gear_Bear_Temp residuo medio: +0.97 degC (baseline annua +0.47; delta +0.50)
  Gen_Bear_Temp residuo medio: +1.90 degC (baseline annua +1.18; delta +0.72)
  HI medio pre-guasto: ~0.74 (zona attenzione; in linea con la media annua 0.73)
  Frazione allarmi pre-guasto: ~0.02, NON superiore al fondo annuo (~0.055)
L'andamento settimanale del residuo gearbox non e' monotono: debole picco intorno all'8 ottobre
(~9 giorni prima del guasto), seguito da calo.
Pattern diagnostico caratteristico:
  - Degradazione PROGRESSIVA e MONOTONA del residuo gearbox
  - Gear_Oil e Gear_Bear entrambi anomali (guasto interno gearbox)
  - Gen_Bear nella norma (esclude guasto generatore)
  - Segnale termico debole, non separabile in modo affidabile dal rumore di fondo tramite il solo
Health Index aggregato
  - Nessun anticipo di rilevazione affidabile: il detector termico non isola questo guasto dalla
variabilita' annuale e stagionale
  - Firma tenue coerente con frattura gabbia planetaria (manifestazione primariamente
vibrazionale, non termica)
Similarità con questo caso: se il pattern attuale mostra
residuo positivo progressivo su Gear_Oil e Gear_Bear
con Gen_Bear nella norma, il pattern e' SIMILE a questo caso.


Diagnosi fisica confermata da ispezione (España et al. 2026):
  L'ispezione post-guasto del gearbox WT8 (corrispondente a T06) ha
  rivelato una SEQUENZA INTERCONNESSA di failure mode, non un guasto
  isolato. La catena di degrado e':
  1. Innesco: frattura della gabbia del cuscinetto Planet 1 (cage failure).
     I rulli, in condizioni di sottocarico, si inclinano fino a 90 gradi
     aumentando lo stress sulla gabbia fino a frattura.
  2. Propagazione: la frattura rilascia detriti metallici nel lubrificante.
  3. Danni secondari indotti dai detriti:
     - Indentazioni superficiali su: Ring gear, Sun pinion, Planet 1 and 2 teeth,
       Planet 2 and 3 bearings, IMS pinion, LSS gear wheel, IMS and LSS bearings
       (non-rotating ends), HSS pinion
     - Usura abrasiva sui denti del Planet 3 gear
     - Fretting corrosion su cuscinetti Planet 2 e 3, IMS, LSS, planet carrier, HSS
     - Scoring sul pignone HSS
  Esito: raccomandazione di shutdown e sostituzione completa del gearbox.
  Rilevanza per AURA-PHM: il pattern termico progressivo rilevato dal NBM
  e' la firma SCADA di questa catena fisica di degrado. Il residuo termico
  e' un indicatore precoce della sequenza, non identifica il singolo modo
  di guasto (vedi DOC1 e nota metodologica finale).
═══════════════════════════════════════════════════════════════
CASO STORICO 2: T07 — GENERATOR BEARINGS DAMAGED
```

```
================================================================
Turbina: T07
Data guasto certificata: 20 Agosto 2017
Tipo guasto: Generator bearings damaged
Nota: Seguito da Generator damaged il 21 Agosto 2017
Fonte: EDP Open Data - opendata-wind-failures-2017
Pattern diagnostico caratteristico:
  - Residuo anomalo ISOLATO su Gen_Bear_Temp_Avg
  - Gear_Oil_Temp e Gear_Bear_Temp nella norma o lievemente elevati
  - Questo pattern DISTINGUE il guasto generatore dal gearbox
  - Progressione tipicamente rapida (giorni, non settimane)
Azione raccomandata per pattern simile:
  - Ispezione generatore prioritaria
  - Non necessariamente fermo gearbox
  - Verificare sistema raffreddamento generatore
================================================================
CASO STORICO 3: T01 — TRANSFORMER FAN DAMAGED
================================================================
Turbina: T01
Data guasto: 11 Agosto 2017
Tipo guasto: Transformer fan damaged
Fonte: EDP Open Data - opendata-wind-failures-2017
Pattern diagnostico:
  - Guasto elettrico/ausiliario, non meccanico gearbox
  - Residui gearbox tipicamente nella norma
  - Possibile impatto su Nac_Temp_Avg (temperatura nacelle)
  - Cross-validation su T01 con modello T06: HI 9% allarmi
    (maggiore sensibilita' cross-turbine normale)
================================================================
DATI DI AFFIDABILITA' DI CAMPO — CONTESTO STATISTICO DI SETTORE
================================================================
Fonte: Carroll et al. (2015), analisi su ~350 turbine eoliche offshore europee.
NOTA: questi sono dati di riferimento OFFSHORE. Le turbine EDP analizzate
(T01, T06, T07, T11) sono ONSHORE. I tassi di guasto offshore sono
generalmente piu' alti degli onshore. I dati Carroll sono usati come ordine
di grandezza e contesto economico del settore, non come valori identici al
caso onshore EDP.


Tassi di guasto (major replacement, costo > 10.000 euro):
  - Gearbox: 0.154 guasti/turbina/anno
  - Generatore: 0.095 guasti/turbina/anno
  - Gearbox + generatore = 95% di tutti i major replacement
Tasso di guasto complessivo turbina: 8.3 guasti/turbina/anno
  (6.2 minor repair, 1.1 major repair, 0.3 major replacement)
Contributo del gearbox al tasso di guasto complessivo: 7.6%
  (generatore 12.1%)


Costi e tempi di riparazione (solo materiali, esclusa manodopera):
  - Gearbox major replacement: ~230.000 euro (il costo piu' alto tra
    tutti i componenti della turbina)
  - Gearbox major replacement: ~231 ore di riparazione
  - Gearbox e hub richiedono il maggior numero di tecnici per riparazione
    (fino a ~20 tecnici nei major replacement piu' complessi)


Failure mode dominanti del gearbox (Carroll 2015 + letteratura):
  La maggior parte dei guasti gearbox e' legata ai cuscinetti dell'albero
  (shaft bearings), con riparazioni molto costose e downtime elevato.
  Cricche possono svilupparsi nei cuscinetti gearbox entro 3 anni di
  operazione della turbina (Stadler & Stubenrauch 2013, citato in Desai et al.).


Rilevanza per AURA-PHM: il gearbox e' tra i componenti piu' critici per
costo e downtime. Questo giustifica la scelta di concentrare il monitoraggio
NBM sulle tre temperature gearbox/generatore (Gear_Oil, Gear_Bear, Gen_Bear)
e di dare priorita' alta agli allarmi su queste componenti.
================================================================
METRICHE DI PERFORMANCE SISTEMA AURA-PHM (validazione empirica)
================================================================
Dataset: EDP SCADA 2016 (training) / 2017 (test)
Turbina principale: T06
Modello: Random Forest NBM multi-target
Performance NBM sul training set 2016:
  Gear_Oil_Temp_Avg: MAE=0.590 degC, R2=0.9718
```

```
  Gear_Bear_Temp_Avg: MAE=0.443 degC, R2=0.9879
  Gen_Bear_Temp_Avg: MAE=1.133 degC, R2=0.9852
Performance NBM sul test set 2017 (blind - dati mai visti):
  Gear_Oil_Temp_Avg: MAE=1.984 degC, R2=0.670
  Gear_Bear_Temp_Avg: MAE=1.551 degC, R2=0.864
  Gen_Bear_Temp_Avg: MAE=4.193 degC, R2=0.792
Distribuzione stati 2017 (detection cieca):
  Normale: 81.3% (29,639 campioni)
  Attenzione: 13.3% (4,836 campioni)
  Allarme L3: 5.5% (1,993 campioni)
Cross-validation T01 (turbina mai usata nel training):
  Allarmi: 9.0% - generalizzazione confermata
Anticipo rilevazione guasto T06 17/10/2017:
  Frazione allarmi pre-guasto (3-16 ott): ~0.02, non superiore al fondo annuo (~0.055)
  Il detector termico non fornisce anticipo affidabile: segnale debole, visibile solo con analisi
mirata per-segnale
```

A.3: DOC3 Operational context of the EDP fleet

```
CONTESTO OPERATIVO TURBINE EDP WIND FARM
Documento tecnico AURA-PHM — Knowledge Base stratificata
Fonte: EDP Wind Farm Open Data 2016 - statistiche calcolate
Wind Farm: EDP Wind Farm 1 (4 turbine: T01, T06, T07, T11)


Fonte normativa certificata:
  BS EN IEC 61400-4:2025 - Design requirements for wind turbine gearboxes


NOTA METODOLOGICA: le specifiche operative e i valori normali sono dati reali
calcolati dal dataset EDP 2016. Le soglie di interpretazione dei residui sono
data-driven dal NBM. I riferimenti IEC ancorano il significato fisico delle
temperature monitorate, ma NON forniscono soglie sui residui (l'IEC definisce
limiti assoluti di shutdown, non soglie di anticipo data-driven).
============================================================
SPECIFICHE OPERATIVE TURBINA
============================================================
Velocità vento cut-in: 3.0 m/s
Velocità vento cut-out: 25.0 m/s
Potenza nominale (rated): ~2000 kW (colonna Grd_Prod_Pwr_Avg)
Risoluzione SCADA: 10 minuti (medie)
Anni operativi osservati: 2016-2017
============================================================
VALORI OPERATIVI NORMALI — TURBINA T06 (anno 2016, training NBM)
============================================================
Statistiche calcolate su 33,078 campioni operativi (filtrati):
Velocità vento:
  Media: 7.88 m/s | Std: 3.61 m/s
  Range operativo tipico: 3.0 - 24.5 m/s
Temperature gearbox in condizioni normali (anno 2016):
  Gear_Oil_Temp_Avg:
    Media: 51.05 degC | Std: 4.44 degC
    Range tipico: 27-61 degC
    Valori attesi dal NBM a diversi carichi:
      Vento 5 m/s (basso carico):  ~47-49 degC
      Vento 9 m/s (carico medio):  ~51-53 degC
      Vento 13 m/s (alto carico):  ~54-58 degC
  Gear_Bear_Temp_Avg:
    Media: 57.15 degC | Std: 5.61 degC
    Range tipico: 28-69 degC
    Valori attesi NBM:
      Vento 5 m/s:  ~53-55 degC
      Vento 9 m/s:  ~57-60 degC
      Vento 13 m/s: ~61-65 degC
  Gen_Bear_Temp_Avg:
    Range tipico 2016: variabile con stagione
    Nota: maggiore variabilita' stagionale rispetto a gearbox


Significato fisico delle temperature monitorate (IEC 61400-4:2025):
  La temperatura operativa del lubrificante e' un parametro importante usato per
  approssimare la viscosita' del lubrificante e lo spessore del film lubrificante,
  e varia all'interno del gearbox (IEC 61400-4:2025 Clause 6.6.6.1). La viscosita'
  del lubrificante alle condizioni operative e' un parametro chiave che influenza
```

```
  il comportamento del gearbox, inclusi i calcoli di rating e le condizioni del
  film lubrificante (Clause 6.6.3). Per questo il monitoraggio della temperatura olio
  (Gear_Oil) e dei cuscinetti (Gear_Bear) e' diagnosticamente significativo: una
  deviazione dal valore atteso segnala un cambiamento nelle condizioni di
  lubrificazione o di carico.


Interpretazione contesto stagionale:
  ESTATE (giu-ago): temperature ambiente +25-35 degC
    → temperature gearbox naturalmente piu' alte
    → residuo NBM piu' rumoroso (maggiore varianza)
  INVERNO (nov-feb): temperature ambiente +5-15 degC
    → temperature gearbox piu' basse
    → residuo NBM piu' stabile e affidabile
    → ATTENZIONE: residuo positivo invernale e' piu' significativo
      perche' non spiegabile da temperatura ambiente
  AUTUNNO (set-nov): periodo critico per dataset 2017
    → transizione termica rapida
    → modello addestrato su 2016 puo' mostrare residui
      sistematicamente positivi per effetto stagionale
    → guasto T06 rilevato in ottobre: residui autunnali
      interpretati come degrado reale, confermato dal failure log
  Nota IEC: la viscosita' del lubrificante a freddo (avvio/idling) e a temperatura
  operativa influenza il comportamento del gearbox; la temperatura bulk lubricant
  va mantenuta almeno 10 K sopra il pour point in avvio e operazione normale
  (IEC 61400-4:2025 Clause 6.6.7.2). Questo spiega perche' l'effetto stagionale
  sulla temperatura ambiente si riflette sui residui NBM.
═══════════════════════════════════════════════════════════════
INTERPRETAZIONE RESIDUI IN CONTESTO OPERATIVO
═══════════════════════════════════════════════════════════════
Un residuo positivo NON indica automaticamente un guasto.
Va interpretato nel contesto operativo.


NOTA SU SOGLIE vs LIMITI ASSOLUTI: le soglie sottostanti sono soglie di ANTICIPO
data-driven dal NBM, espresse come deviazione (residuo) dal comportamento atteso.
Sono distinte dai limiti assoluti di shutdown IEC 61400-4:2025 (bulk lubricant
85 degC, rolling bearing 95 degC), che agiscono su temperature assolute molto piu'
alte. Il NBM anticipa il degrado mentre la temperatura assoluta e' ancora nella
norma operativa.


Residuo Gear_Oil < 3 degC:
  → Variazione normale entro 1 sigma del modello
  → Nessuna azione richiesta
Residuo Gear_Oil 3-8 degC persistente per > 24h:
  → Zona attenzione
  → Monitorare evoluzione nelle successive 48h
  → Verificare condizioni ambientali (ondata di calore?)
Residuo Gear_Oil > 8 degC persistente per > 12h:
  → Anomalia termica significativa
  → Non spiegabile solo da variazioni ambientali
  → Attivare L3 (RAG-LLM) per diagnosi
Residuo Gear_Oil > 15 degC:
  → Anomalia critica
  → Azione immediata raccomandata
Residuo negativo Gear_Oil < -5 degC persistente:
  → Inusuale: temperatura reale piu' bassa del previsto
  → Possibile: sensore difettoso, turbina parzialmente
    ferma, riduzione carico non catturata dal modello
  → Verificare operativita' della turbina
```

A.4: DOC4 Urgency and action guide

```
GUIDA AZIONI RACCOMANDATE — SISTEMA AURA-PHM
Protocollo di risposta per operatori di parchi eolici — Knowledge Base stratificata
Fonte: Pandit, Astolfi, Durazo Cardenas (2023) Energies 16, 1654
       España et al. (2026) Results in Engineering 30, 109915
       Castellani et al. (2024) Energy Reports 12, 750-761
       Carroll et al. (2015) Wind Energy - O&M cost analysis


Fonte normativa certificata:
       BS EN IEC 61400-4:2025 - Design requirements for wind turbine gearboxes
```

NOTA METODOLOGICA: i livelli di urgenza, le tempistiche di intervento e il
protocollo di diagnosi differenziale sono logica operativa del sistema AURA-PHM,
calibrata sui dati EDP. I riferimenti IEC ancorano le procedure di verifica
(ispezione, campionamento lubrificante) allo standard. Le soglie HI e i tempi
di intervento sono scelte progettuali del sistema, non prescrizioni IEC.

LIVELLI DI URGENZA IN BASE ALL'HEALTH INDEX

LIVELLO VERDE — HI >= 0.60 (NORMALE)
  Stato: Turbina operativa in condizioni normali
  Azione: Nessuna azione immediata richiesta
  Monitoraggio: Standard (ogni turno)
  Nota: Continuare operazione normale
LIVELLO GIALLO — 0.40 <= HI < 0.60 (ATTENZIONE)
  Stato: Anomalia termica rilevata, non critica
  Azione obbligatoria entro 48 ore:
    1. Analisi visiva remota dati SCADA (tendenza ultimi 7 giorni)
    2. Verifica condizioni ambientali (escludere cause esterne)
    3. Schedulare ispezione tecnica entro 7 giorni
    4. Aumentare frequenza analisi olio (se non eseguita < 2 sett.)
  Monitoraggio: Ravvicinato (ogni 6 ore)
  Continuare operazione: SI, con monitoraggio aumentato
LIVELLO ROSSO — HI < 0.40 (ALLARME — attiva L3)
  Stato: Anomalia termica significativa — guasto probabile
  Azione obbligatoria entro 24 ore:
    1. Notifica immediata al responsabile manutenzione
    2. Analisi lubrificante urgente (cercare particelle metalliche)
    3. Ispezione tecnica gearbox entro 48-72 ore
    4. Valutare riduzione carico operativo (non fermo immediato
       salvo residui > 20 degC o HI < 0.10)
    5. Documentare nel sistema di manutenzione
  Monitoraggio: Continuo (ogni campione SCADA = ogni 10 minuti)
  Continuare operazione: CONDIZIONALE
    - HI 0.20-0.40: continuare con riduzione carico
    - HI < 0.20: valutare fermo per ispezione immediata
LIVELLO CRITICO — HI < 0.20 (CRITICO)
  Stato: Guasto avanzato probabile
  Azione: Considerare fermo turbina per ispezione immediata
  Giustificazione economica: Il costo di un fermo pianificato
    (circa 1-2 giorni) e' significativamente inferiore al costo
    di un guasto catastrofico al gearbox. Una sostituzione completa
    del gearbox (major replacement) ha un costo medio materiali di
    circa 230.000 euro ed un tempo di riparazione di circa 231 ore,
    il piu' alto tra tutti i componenti della turbina
    (Carroll et al. 2015, dati di campo offshore).

SUPPORTO IEC ALLE AZIONI DI VERIFICA

Le azioni di verifica raccomandate ai livelli GIALLO, ROSSO e CRITICO sono
supportate dai requisiti di servizio e monitoraggio di IEC 61400-4:2025:
  - Analisi del lubrificante: il monitoraggio delle caratteristiche del
    lubrificante puo' essere giudicato su limiti di allarme assoluti, su
    cambiamenti di trend, o una combinazione dei due, con criteri di
    accettazione e azioni raccomandate (Clause 6.6.8). Questo supporta
    l'uso del trend del residuo NBM come innesco delle azioni.
  - Ricerca di particelle metalliche: i tappi magnetici (magnetic plugs)
    servono a rilevare le particelle di usura e vanno posti in area
    accessibile e in regione di circolazione del lubrificante (Clause 6.6.12).
  - Ispezione endoscopica: il gearbox e' progettato per accomodare
    l'ispezione con video probe (boroscopio) di tutti gli alberi, ingranaggi
    e cuscinetti (Clause 9.2).
  - Schedule di servizio: la documentazione include lo schedule di servizio
    con istruzioni per ispezione di alberi/ingranaggi/cuscinetti e
    campionamento del lubrificante (Clause 9.3).

PROTOCOLLO DI DIAGNOSI DIFFERENZIALE

Per distinguere il tipo di guasto dal pattern di residui.
Riferimento incrociato: i pattern sottostanti corrispondono ai modi di guasto
classificati IEC nel documento FMEA (DOC1).

Pattern A — Guasto gearbox (cuscinetti/olio):
  Gear_Oil anomalo E Gear_Bear anomalo
  Gen_Bear nella norma

```
  → Problema interno al gearbox
  → Modi di guasto IEC associati: cage fracture (Cat. A2), abrasive wear,
    fretting, scuffing (Cat. B) - vedi DOC1
  → Azioni: analisi olio, ispezione gearbox
Pattern B — Guasto generatore:
  Gen_Bear anomalo ISOLATO
  Gear_Oil e Gear_Bear nella norma o lievemente elevati
  → Problema al generatore, non al gearbox
  → Modo di guasto IEC associato: overheating cuscinetti generatore (Cat. B) - DOC1
  → Azioni: ispezione generatore, sistema raffreddamento
Pattern C — Anomalia diffusa:
  Tutti e tre i target anomali simultaneamente
  → Possibile: effetto ambientale (ondata calore),
    problema sistemico, o guasto avanzato multi-componente
  → Azioni: verificare prima cause ambientali,
    poi procedere come Pattern A se persiste
Pattern D — Picco isolato senza persistenza:
  HI scende sotto soglia per 1-2 campioni poi risale
  → Probabile: rumore sensoriale, transiente operativo
  → Azioni: monitorare per 24h prima di escalare
═══════════════════════════════════════════════════════════
TRASFERIMENTO CONOSCENZA TRA TURNI
═══════════════════════════════════════════════════════════
Il sistema AURA-PHM genera automaticamente un briefing di turno
a fine di ogni turno (06:00, 14:00, 22:00).
Il briefing include:
  - Riepilogo allarmi rilevati nel turno
  - Evoluzione HI per turbina
  - Azioni intraprese e loro esito
  - Anomalie aperte da trasferire al turno successivo
  - Pattern di residuo in evoluzione da monitorare
Questo meccanismo risponde al gap documentato in letteratura
sul knowledge management in manutenzione: la perdita di conoscenza
tra turni e' una delle cause di ritardo nella risposta a guasti
incipienti nei parchi eolici (Human-in-the-Loop XAI for Predictive
Maintenance, 2025).
```

Appendix B: Prompts and generation configuration

The instructions given to the language model in the diagnostic reasoning layer (Section 4.4.3), reproduced verbatim. Generation uses gpt-4o-mini in constrained JSON mode, at temperature 0.1 for diagnosis and 0.2 for the shift briefing. The system prompt fixes the binding rules on residual sign, the anti-contradiction guardrails, and the response schema.

Response schema (Italian key, English meaning): diagnosi_probabile = probable diagnosis (1-2 sentences); causa_radice = root cause, with cited source; pattern_rilevato = detected residual pattern; urgenza = urgency (IMMEDIATA / 24h / 48-72h / 7giorni / Monitoraggio); azione_raccomandata = recommended action, with specific timing; fonte_principale = principal source cited; limitazione = stated limitation.

B.1: Diagnostic system prompt and user template

```
Sei AURA-PHM, sistema esperto di diagnostica per turbine eoliche.
Analizza i dati SCADA e i documenti tecnici per fornire diagnosi precise.


Regole:
1. Basa ogni affermazione sui documenti tecnici forniti
2. Cita le fonti esplicitamente
3. Usa "probabile" o "possibile" - mai certezze assolute
4. Indica sempre l'urgenza dell'azione raccomandata
5. La diagnosi finale richiede conferma dell'operatore
```

```
INTERPRETAZIONE DEL SEGNO DEI RESIDUI (regola permanente, vincolante):
Il residuo è (valore reale − valore atteso). Il SEGNO è diagnostico e va sempre rispettato:
- Residuo POSITIVO = temperatura SOPRA l'atteso → possibile surriscaldamento / guasto
  in sviluppo (attrito, degrado lubrificante, sovraccarico).
- Residuo NEGATIVO = temperatura SOTTO l'atteso → NON è surriscaldamento. Indica
  più probabilmente un transitorio operativo, turbina parzialmente ferma / a basso
  carico, oppure un'anomalia/deriva del sensore.
- Il valore assoluto è usato solo per la severità della detection, MAI per
  l'interpretazione fisica della causa. L'urgenza alta resta giustificata dal dato
  anomalo: NON minimizzarla né dichiarare la turbina sana.

Leggi PRIMA il segno del residuo dominante, indicato nella riga "PATTERN RILEVATO"
dei dati, e applica ESCLUSIVAMENTE la regola del segno corrispondente. Non confondere
i due casi e non applicare la regola del segno opposto.

GUARDRAIL ANTI-CONTRADDIZIONE (vincolante): il segno che dichiari nella diagnosi DEVE
coincidere con il segno del residuo dominante riportato nella riga "PATTERN RILEVATO".
Se quella riga dice "residui positivi dominanti" NON affermare mai "residuo negativo"
(e viceversa). Diagnosi, causa radice, pattern rilevato e azione raccomandata devono
essere coerenti tra loro e con il segno del dato.

REGOLA VINCOLANTE quando i residui dominanti (i maggiori in valore assoluto) sono POSITIVI:
1. "diagnosi_probabile" DEVE aprire con un possibile surriscaldamento / guasto del
   componente in sviluppo. NON citare ipotesi di residuo negativo, transitorio o
   turbina ferma.
2. "azione_raccomandata": ispezione prioritaria del componente coinvolto e verifica
   del sistema di raffreddamento, con la tempistica adeguata all'urgenza.

REGOLA VINCOLANTE quando i residui dominanti (i maggiori in valore assoluto) sono NEGATIVI:
1. "diagnosi_probabile" DEVE aprire con l'ipotesi NON-guasto come spiegazione più
   probabile: transitorio operativo, turbina parzialmente ferma / a basso carico,
   oppure anomalia / deriva del sensore. NON aprire con un guasto del componente
   né con un surriscaldamento.
2. "azione_raccomandata" DEVE essere DIAGNOSTICA e di VERIFICA, in quest'ordine:
   (a) verificare l'integrità e la taratura del sensore del componente coinvolto;
   (b) confermare lo stato operativo della turbina — accertare che non sia ferma,
       in arresto o a basso carico.
   Solo se queste verifiche escludono transitorio e anomalia sensore si passa
   all'ispezione meccanica del componente. NON raccomandare come prima azione
   l'ispezione meccanica / del generatore.
3. Richiama esplicitamente il pattern del DOC3 ("residuo negativo persistente →
   sensore difettoso / turbina parzialmente ferma"). NON applicare pattern di
   surriscaldamento a una macchina più fredda dell'atteso.

Rispondi ESCLUSIVAMENTE con un oggetto JSON valido:
{
  "diagnosi_probabile": "stringa 1-2 frasi",
  "causa_radice": "stringa con fonte citata",
  "pattern_rilevato": "stringa descrizione pattern residui",
  "urgenza": "IMMEDIATA / 24h / 48-72h / 7giorni / Monitoraggio",
  "azione_raccomandata": "stringa con tempistica specifica",
  "fonte_principale": "stringa citazione documento",
  "limitazione": "stringa cosa non determinabile con certezza"
}
Nessun testo fuori dal JSON. Nessun markdown. Solo JSON valido.
```

User template (dati_str and contesto_kb are substituted at call time):

```
Analizza questa anomalia termica:

{dati_str}

Documenti tecnici disponibili:
{contesto_kb}

Genera la diagnosi in formato JSON.
```

B.2: Shift-briefing prompt

```
Sei AURA-PHM. Genera un briefing professionale per il tecnico
del turno successivo.

{riepilogo}
{contesto_storico}

Note dell'operatore lasciate durante il turno:
{note_turno}

Anomalie attualmente aperte (cumulativa su tutti i turni) (lista VERIFICATA, fornita dal sistema — verrà inserita
automaticamente nel briefing; NON riprodurla e NON creare una tua sezione "ANOMALIE APERTE"):
{blocco_anomalie}

Struttura il briefing con (NON includere una sezione "ANOMALIE APERTE": la inserisce il sistema):
1. SITUAZIONE ATTUALE (2-3 frasi)
2. AZIONI PRIORITARIE per il prossimo turno (puoi riferirti alle anomalie aperte elencate sopra)
3. TREND (confronto con turni precedenti se disponibile)
4. NOTE DALL'OPERATORE (riporta integralmente le osservazioni lasciate dall'operatore durante il
turno, se ce ne sono; altrimenti scrivi "Nessuna nota di passaggio dall'operatore")

Max 250 parole. Stile diretto e operativo.
```

Appendix C: Evaluation set and per-question RAGAS results

The twenty-one curated question-reference pairs used for the quantitative evaluation of the diagnostic layer (Section 5.3), with the RAGAS scores for each question. The body reports the aggregate means (faithfulness 0.877, answer relevance 0.707, context relevance 0.159); this appendix gives the full per-question breakdown, reproduced from the recorded evaluation run, in keeping with the honest-evaluation commitment of Gap 4. Questions are in the working language (Italian).

#	**Domanda (IT)**	**Faith.**	**Ans.rel.**	**Ctx.rel.**	**RAGAS**
1	Cosa significa un residuo positivo di 14 gradi sulla temperatura dell olio del g	0.9091	0.8341	0.0833	0.6088
2	Quali sono le azioni raccomandate quando l HI scende sotto 0.40?	0.5	0.6155	0.4286	0.5147
3	Come si distingue un guasto al gearbox da un guasto al generatore dai dati SCADA	0.8	0.626	0.125	0.517
4	Cosa è successo alla turbina T06 il 17 ottobre 2017?	0.8571	0.7236	0.1111	0.5639
5	Qual è il range operativo normale della temperatura olio gearbox per la turbina	1.0	0.7921	0.0909	0.6277
6	Quali sono i sintomi SCADA della frattura della gabbia del cuscinetto planetario	1.0	0.6723	0.1333	0.6019
7	Cosa causa l usura abrasiva degli ingranaggi planetari e come si rileva da SCADA	0.9167	0.6993	0.125	0.5803
8	Cosa è il Health Index e come viene calcolato?	0.8462	0.7176	0.1364	0.5667
9	Quali azioni devo eseguire con HI tra 0.40 e 0.60?	1.0	0.6621	0.3571	0.6731
10	Come influisce la stagione invernale sull interpretazione dei residui NBM?	1.0	0.5417	0.1111	0.5509
11	Cosa contiene il sistema di knowledge base di AURA PHM?	0.8125	0.5772	0.3125	0.5674

12	Cosa sono i dati SCADA e perché sono importanti per la manutenzione predittiva?	1.0	0.7679	0.2	0.656
13	Qual è la frequenza di campionamento dei dati SCADA del wind farm EDP?	0.5	0.8562	0.0833	0.4798
14	Cosa indica un picco isolato di HI sotto soglia senza persistenza?	1.0	0.7273	0.0714	0.5996
15	Quali turbine sono presenti nel wind farm EDP del dataset?	1.0	0.8113	0.0909	0.6341
16	Cosa è il fretting corrosion nei cuscinetti e come si rileva?	0.8333	0.7247	0.1429	0.567
17	Quali componenti del gearbox di T06 erano anomali nelle settimane prima del guas	0.5556	0.7428	0.1538	0.4841
18	Come funziona il Normal Behaviour Model di AURA PHM?	0.9333	0.6073	0.4444	0.6617
19	Cosa si intende per separazione temporale nel training del modello?	1.0	0.7222	0.0	0.5741
20	Qual è la severità del guasto per frattura della gabbia cuscinetto planetario?	1.0	0.7895	0.1333	0.6409
21	Cosa fa il sistema AURA PHM quando HI scende sotto 0.20?	0.8571	0.5928	0.0833	0.5111

Appendix D: Full diagnostic and classification traces

The complete recorded traces summarised in Tables 5.2 and 5.3. For each diagnostic case: the health index and residuals that raised the alarm, the retrieval query, every retrieved chunk with its source and score, and the full generated diagnosis. For each classification case: the query, the expected and assigned category, the margin, and the pool consulted. These are the raw artefacts behind Section 5.4.

D.1: End-to-end diagnostic traces (Section 5.4.2)

```
Case 1  gearbox-oil dominant
timestamp: 2017-11-10 13:40:00+00:00   HI: 0.2167   componente: Gear_Oil_Temp_Avg
urgenza  deterministica: 24h  |  LLM: IMMEDIATA
residui: {'Gear_Oil_Temp_Avg': 10.79, 'Gear_Bear_Temp_Avg': 5.44, 'Gen_Bear_Temp_Avg': 8.51}
retrieval query: wind turbine gearbox oil fault Gear_Oil_Temp_Avg Allarme health index 0.217
Gearbox Oil Temp residual 10.8degC Gearbox Bearing Temp residual 5.4degC Generator Bearing Temp
residual 8.5degC thermal anomaly temperature diagnosis action
chunk recuperati:
  - DOC2_FaultArchive_EDP.txt #3 (tecnica, score 1.0)
      all'8 ottobre (~9 giorni prima del guasto), seguito da calo. Pattern diagnostico
caratteristico:   - Degradazione PROGRESSIVA e MONOTONA del residuo g
  - DOC3_ContestoOperativo_EDP.txt #10 (tecnica, score 0.9549)
      er > 12h:   → Anomalia termica significativa   → Non spiegabile solo da variazioni
ambientali   → Attivare L3 (RAG-LLM) per diagnosi Residuo Gear_Oil
  - DOC2_FaultArchive_EDP.txt #8 (tecnica, score 0.9368)
      stico:   - Residuo anomalo ISOLATO su Gen_Bear_Temp_Avg   - Gear_Oil_Temp e Gear_Bear_Temp
nella norma o lievemente elevati   - Questo pattern DISTING
  - BRIEFING_1_2017-10-08 07-50.txt #0 (operativa, score 0.6818)
      BRIEFING TURNO 1 — 2017-10-08 07:50 ANOMALIE APERTE:   - T06 — Gear_Oil_Temp_Avg — HI=0.316
— 48-72h — CONFERMATO  **BRIEFING PER IL TECNICO DEL TURNO
diagnosi generata:
  diagnosi_probabile: Possibile surriscaldamento / guasto del componente in sviluppo, in
particolare del sistema di lubrificazione del gearbox.
  causa_radice: Il residuo positivo di 10.79 degC per il Gear_Oil_Temp indica un'anomalia termica
significativa, suggerendo un guasto interno al gearbox (FONTE: DOC2_FaultArchive_EDP.txt).
  pattern_rilevato: Residui positivi dominanti — temperatura sopra l'atteso per Gear_Oil_Temp.
  urgenza: 24h
  azione_raccomandata: Ispezione prioritaria del sistema di lubrificazione del gearbox e verifica
del sistema di raffreddamento, da effettuare senza indugi.
  fonte_principale: FONTE: DOC2_FaultArchive_EDP.txt
  limitazione: Non è possibile determinare con certezza la causa esatta del surriscaldamento senza
un'ispezione diretta.

Case 2  generator-bearing isolated
timestamp: 2017-03-03 21:50:00+00:00   HI: 0.3619   componente: Gen_Bear_Temp_Avg
```

```
urgenza  deterministica: 7 giorni  |  LLM: IMMEDIATA
residui: {'Gear_Oil_Temp_Avg': 3.36, 'Gear_Bear_Temp_Avg': 3.6, 'Gen_Bear_Temp_Avg': 31.16}
retrieval query: wind turbine generator bearing fault Gen_Bear_Temp_Avg Allarme health index 0.362
Gearbox Oil Temp residual 3.4degC Gearbox Bearing Temp residual 3.6degC Generator Bearing Temp
residual 31.2degC thermal anomaly temperature diagnosis action
chunk recuperati:
  - DOC2_FaultArchive_EDP.txt #3 (tecnica, score 1.0)
      all'8 ottobre (~9 giorni prima del guasto), seguito da calo. Pattern diagnostico
caratteristico:   - Degradazione PROGRESSIVA e MONOTONA del residuo g
  - DOC2_FaultArchive_EDP.txt #8 (tecnica, score 0.9576)
      stico:   - Residuo anomalo ISOLATO su Gen_Bear_Temp_Avg   - Gear_Oil_Temp e Gear_Bear_Temp
nella norma o lievemente elevati   - Questo pattern DISTING
  - DOC2_FaultArchive_EDP.txt #15 (tecnica, score 0.956)
       R2=0.9852 Performance NBM sul test set 2017 (blind - dati mai visti):   Gear_Oil_Temp_Avg:
MAE=1.984 degC, R2=0.670   Gear_Bear_Temp_Avg: MAE=1.551 d
  - BRIEFING_4_2017-10-09 08-40.txt #1 (operativa, score 0.6532)
      e un'anomalia del sensore per il componente Generator Bearing. Entrambe le anomalie
richiedono attenzione nei prossimi 48-72 ore.  2. **AZIONI PRIORIT
diagnosi generata:
  diagnosi_probabile: Possibile surriscaldamento / guasto del componente in sviluppo, in
particolare del cuscinetto del generatore, dato il residuo positivo dominante di +31.16 degC.
  causa_radice: Il residuo anomalo isolato su Gen_Bear_Temp_Avg suggerisce un guasto potenziale
del generatore, come indicato in DOC2_FaultArchive_EDP.txt.
  pattern_rilevato: Residui positivi dominanti — temperatura sopra l'atteso (residuo dominante
Generator Bearing Temp: +31.16 degC).
  urgenza: 7giorni
  azione_raccomandata: Ispezione prioritaria del generatore e verifica del sistema di
raffreddamento, da effettuare senza indugi.
  fonte_principale: DOC2_FaultArchive_EDP.txt
  limitazione: Non è possibile determinare con certezza se il guasto sia già avvenuto o se sia in
fase di sviluppo.

Case 3a  gearbox-oil, deep alarm
timestamp: 2017-12-12 13:10:00+00:00   HI: 0.0779   componente: Gear_Oil_Temp_Avg
urgenza  deterministica: immediata  |  LLM: IMMEDIATA
residui: {'Gear_Oil_Temp_Avg': 11.46, 'Gear_Bear_Temp_Avg': 7.18, 'Gen_Bear_Temp_Avg': 12.21}
retrieval query: wind turbine gearbox oil fault Gear_Oil_Temp_Avg Allarme health index 0.078
Gearbox Oil Temp residual 11.5degC Gearbox Bearing Temp residual 7.2degC Generator Bearing Temp
residual 12.2degC thermal anomaly temperature diagnosis action
chunk recuperati:
  - DOC2_FaultArchive_EDP.txt #3 (tecnica, score 1.0)
      all'8 ottobre (~9 giorni prima del guasto), seguito da calo. Pattern diagnostico
caratteristico:   - Degradazione PROGRESSIVA e MONOTONA del residuo g
  - DOC3_ContestoOperativo_EDP.txt #10 (tecnica, score 0.9561)
      er > 12h:   → Anomalia termica significativa   → Non spiegabile solo da variazioni
ambientali   → Attivare L3 (RAG-LLM) per diagnosi Residuo Gear_Oil
  - DOC2_FaultArchive_EDP.txt #8 (tecnica, score 0.9319)
      stico:   - Residuo anomalo ISOLATO su Gen_Bear_Temp_Avg   - Gear_Oil_Temp e Gear_Bear_Temp
nella norma o lievemente elevati   - Questo pattern DISTING
  - BRIEFING_1_2017-10-08 07-50.txt #0 (operativa, score 0.6828)
      BRIEFING TURNO 1 — 2017-10-08 07:50 ANOMALIE APERTE:   - T06 — Gear_Oil_Temp_Avg — HI=0.316
— 48-72h — CONFERMATO  **BRIEFING PER IL TECNICO DEL TURNO
diagnosi generata:
  diagnosi_probabile: Possibile surriscaldamento / guasto del componente in sviluppo, in
particolare del sistema di lubrificazione del gearbox.
  causa_radice: Il residuo positivo dominante di 11.46 degC indica un'anomalia termica
significativa, suggerendo un guasto interno al gearbox (FONTE: DOC2_FaultArchive_EDP.txt).
  pattern_rilevato: Residui positivi dominanti — temperatura sopra l'atteso (residuo dominante
Gearbox Oil Temp: +11.46 degC).
  urgenza: IMMEDIATA
  azione_raccomandata: Ispezione prioritaria del sistema di lubrificazione del gearbox e verifica
del sistema di raffreddamento, da effettuare senza indugi.
  fonte_principale: FONTE: DOC2_FaultArchive_EDP.txt
  limitazione: Non è possibile determinare con certezza la causa esatta del surriscaldamento senza
un'ispezione diretta.

Case 3b  gearbox-oil, borderline alarm
timestamp: 2017-12-12 18:00:00+00:00   HI: 0.3921   componente: Gear_Oil_Temp_Avg
urgenza  deterministica: 7 giorni  |  LLM: IMMEDIATA
residui: {'Gear_Oil_Temp_Avg': 5.6, 'Gear_Bear_Temp_Avg': 3.48, 'Gen_Bear_Temp_Avg': 4.39}
retrieval query: wind turbine gearbox oil fault Gear_Oil_Temp_Avg Allarme health index 0.392
Gearbox Oil Temp residual 5.6degC Gearbox Bearing Temp residual 3.5degC Generator Bearing Temp
residual 4.4degC thermal anomaly temperature diagnosis action
chunk recuperati:
  - DOC2_FaultArchive_EDP.txt #3 (tecnica, score 1.0)
```

```
      all'8 ottobre (~9 giorni prima del guasto), seguito da calo. Pattern diagnostico
caratteristico:   - Degradazione PROGRESSIVA e MONOTONA del residuo g
  - DOC3_ContestoOperativo_EDP.txt #10 (tecnica, score 0.9565)
      er > 12h:   → Anomalia termica significativa   → Non spiegabile solo da variazioni
ambientali   → Attivare L3 (RAG-LLM) per diagnosi Residuo Gear_Oil
  - DOC2_FaultArchive_EDP.txt #8 (tecnica, score 0.9297)
      stico:   - Residuo anomalo ISOLATO su Gen_Bear_Temp_Avg   - Gear_Oil_Temp e Gear_Bear_Temp
nella norma o lievemente elevati   - Questo pattern DISTING
  - BRIEFING_1_2017-10-08 07-50.txt #0 (operativa, score 0.679)
      BRIEFING TURNO 1 — 2017-10-08 07:50 ANOMALIE APERTE:   - T06 — Gear_Oil_Temp_Avg — HI=0.316
— 48-72h — CONFERMATO  **BRIEFING PER IL TECNICO DEL TURNO
diagnosi generata:
  diagnosi_probabile: Possibile surriscaldamento / guasto del componente in sviluppo, in
particolare del sistema di lubrificazione del gearbox.
  causa_radice: Il residuo positivo di 5.60 degC per il Gear_Oil_Temp suggerisce un'anomalia
termica significativa, non spiegabile solo da variazioni ambientali (FONTE:
DOC3_ContestoOperativo_EDP.txt).
  pattern_rilevato: Residui positivi dominanti — temperatura sopra l'atteso (residuo dominante
Gearbox Oil Temp: +5.60 degC).
  urgenza: 7giorni
  azione_raccomandata: Ispezione prioritaria del sistema di lubrificazione del gearbox e verifica
del sistema di raffreddamento, da effettuare senza indugi.
  fonte_principale: FONTE: DOC2_FaultArchive_EDP.txt
  limitazione: Non è possibile determinare con certezza la causa esatta del surriscaldamento senza
un'ispezione diretta.
```

English translation of the Case 1 generated diagnosis (representative). Probable diagnosis: possible developing overheating or fault of the component, in particular the gearbox lubrication system. Root cause: the positive residual of 10.79 degC on Gear_Oil_Temp indicates a significant thermal anomaly, suggesting an internal gearbox fault (source: DOC2_FaultArchive_EDP.txt). Detected pattern: dominant positive residuals, temperature above expectation for Gear_Oil_Temp. Urgency: 24h. Recommended action: priority inspection of the gearbox lubrication system and verification of the cooling system, to be carried out without delay. Principal source: DOC2_FaultArchive_EDP.txt. Limitation: the exact cause of the overheating cannot be determined with certainty without a direct inspection.

D.2: Classification routing cases (Section 5.4.1)

```
query: Il caso del gearbox è stato chiuso o è passato al turno successivo?
attesa: operativa  ->  assegnata: diagnosi   (corretta: False)
margine: 0.0004   fallback: True margine<0.05
pool consultato: misto (tecnico+operativo)
similarita: {'tecnica': 0.5752, 'operativa': 0.5748, 'diagnosi': 0.5712}

query: Devo aprire un intervento o l'anomalia è già stata chiusa nel turno scorso?
attesa: operativa  ->  assegnata: operativa   (corretta: True)
margine: 0.1845   fallback: False
pool consultato: operativa
similarita: {'tecnica': 0.3592, 'operativa': 0.6538, 'diagnosi': 0.4692}

query: Quale standard supporta la diagnosi di questo modo di guasto?
attesa: tecnica  ->  assegnata: diagnosi   (corretta: False)
margine: 0.0422   fallback: True margine<0.05
pool consultato: misto (tecnico+operativo)
similarita: {'tecnica': 0.5696, 'operativa': 0.3817, 'diagnosi': 0.5274}

query: Qual è il limite di shutdown dell'olio secondo lo standard, non il residuo osservato?
attesa: tecnica  ->  assegnata: diagnosi   (corretta: False)
margine: 0.0316   fallback: True margine<0.05
pool consultato: misto (tecnico+operativo)
similarita: {'tecnica': 0.5214, 'operativa': 0.3548, 'diagnosi': 0.4897}

query: Questo residuo indica un guasto reale del cuscinetto o un transitorio, al di là di cosa
dice lo standard?
attesa: diagnosi  ->  assegnata: diagnosi   (corretta: True)
margine: 0.0113   fallback: True margine<0.05
```

```
pool consultato: misto (tecnico+operativo)
similarita: {'tecnica': 0.5729, 'operativa': 0.3982, 'diagnosi': 0.5616}
```

Appendix E: Audit trail and shift-continuity artefacts

The traceability substrate (Section 4.5.1) and the continuity mechanism (Section 4.5.3). Every diagnosis is recorded as a 17-field row; the schema is given below, then two real rows, then one complete generated shift briefing from the 8-9 October 2017 sequence analysed in Section 5.5.

E.1: Audit-trail record schema (17 fields)

Field	Meaning
`timestamp`	system time of the diagnosis
`turbina`	monitored asset
`timestamp_scada`	timestamp of the originating SCADA sample
`HI`	Health Index of the sample
`stato`	discrete asset state
`componente_anomalo`	most anomalous component
`residuo_max`	magnitude of that component's residual
`diagnosi`	probable diagnosis (generated)
`urgenza`	deterministic urgency (authoritative)
`urgenza_llm`	model-generated urgency (observational)
`azione`	recommended action (generated)
`fonte`	principal source cited
`feedback`	operator outcome: PENDING / CONFERMATO / CORRETTO
`nota_operatore`	free-text operator note
`tokens_usati`	token cost of the generation
`retrieval_mode`	retrieval mode (semantic / lexical / hybrid)
`chunks_usati`	per-fragment provenance: source, position, score

E.2: Sample audit-trail rows

```
timestamp | turbina | timestamp_scada | HI | stato | componente_anomalo | residuo_max | diagnosi | urgenza |
urgenza_llm | azione | fonte | feedback | nota_operatore | tokens_usati | retrieval_mode | chunks_usati
2026-06-11T16:56:47.426006 | T06 | 2017-10-08 01:20:00+00:00 | 0.3163 | Allarme | Gear_Oil_Temp_Avg | 7.06 |
Possibile surriscaldamento / guasto del componente in svil | 48-72h | IMMEDIATA | Ispezione prioritaria del
sistema di lubrificazione del ge | FONTE: DOC2_FaultArchive_EDP.txt | CONFERMATO |  | 2149 | semantic |
[{"fonte": "DOC2_FaultArchive_EDP.txt", "chunk_id": 3, "sc
2026-06-11T16:59:31.514317 | T06 | 2017-10-08 01:30:00+00:00 | 0.3042 | Allarme | Gen_Bear_Temp_Avg | 6.02 |
Possibile surriscaldamento / guasto del componente in svil | 48-72h | IMMEDIATA | Ispezione prioritaria del
cuscinetto del generatore e veri | DOC2_FaultArchive_EDP.txt | CORRETTO | Verificato sui trend, falso allarme da
transitorio | 2256 | semantic | [{"fonte": "DOC2_FaultArchive_EDP.txt", "chunk_id": 3, "sc
```

E.3: Example generated shift briefing (8-9 October 2017)

```
TURNO 1  —  2017-10-08 07:50


ANOMALIE APERTE:
  - T06 — Gear_Oil_Temp_Avg — HI=0.316 — 48-72h — CONFERMATO
```

BRIEFING PER IL TECNICO DEL TURNO SUCCESSIVO

SITUAZIONE ATTUALE:
Durante il turno attuale, sono stati registrati un totale di 2 allarmi per la turbina T06, con un HI minimo rilevato di 0.304. L'anomalia aperta riguarda un possibile surriscaldamento del sistema di lubrificazione del gearbox, confermata con un'urgenza di 48-72 ore.

AZIONI PRIORITARIE:
È fondamentale monitorare attentamente la turbina T06 e verificare l'andamento della temperatura dell'olio del gearbox. Si consiglia di eseguire controlli approfonditi sul sistema di lubrificazione e sul cuscinetto del generatore, in particolare a causa del residuo positivo di +6.02 °C. Assicurarsi di documentare eventuali ulteriori anomalie e aggiornare il sistema di monitoraggio.

TREND:
Rispetto ai turni precedenti, l'HI di T06 ha mostrato una leggera fluttuazione, con l'ultimo valore di 0.316 che è stato confermato come anomalia. È importante continuare a monitorare i trend per identificare eventuali pattern o anomalie ricorrenti.

NOTE DALL'OPERATORE:
T06 (HI=0.304): Verificato sui trend, falso allarme da transitorio.