Integrating Machine Learning Models and an AI Agent for Decision Support in Engineer-to-Order Manufacturing

Luís David Araújo da Costa

Master's Thesis

FEUP Advisor: Prof. Américo Azevedo

Master's in Industrial Management & Engineering

2026-06-25

Abstract

Engineer-to-order manufacturers must plan projects under substantial uncertainty: operation durations are difficult to estimate, shop-floor capacity is constrained, procurement delays can reshape feasible schedules, and external material-market disruptions can alter execution risk. This thesis develops and evaluates an integrated decision-support framework on real industrial data. The framework combines three predictive and planning components: a two-stage model for manufacturing-order duration prediction, a discrete-time shop-floor simulator that converts duration predictions into dependency- and procurement-aware project plans, and nickel and copper disruption-forecasting pipelines formulated around alert episodes. These components are orchestrated by a tool-using large language model agent, built on a locally hostable open-weight model, which grounds numerical claims in tool outputs and translates uncertainty into natural language.

On held-out data, the duration pipeline predicts realised order durations with a mean absolute error of 1.15 working days ($R^2 = 0.71$). The commodity pipelines produce disruption alerts that improve on a momentum benchmark for both metals. An informal evaluation with managers found the agent's combined outputs clear and actionable, with trust and capability scope emerging as the main reservations. These results indicate that engineer-to-order decision support becomes more operationally useful when prediction, interpretation, and cross-model reasoning are treated as a single design problem rather than as separate tasks.

Keywords: engineer-to-order manufacturing; decision support systems; manufacturing order duration prediction; commodity disruption forecasting; supply chain risk; large language models; tool-using AI agents

Acknowledgements

To my family and friends, for their constant support and encouragement over the course of this project.

To Artur, for his invaluable mentorship, and for cultivating creativity and freedom throughout the project; to Pedro, for his companionship over the whole process; and to Professor Américo, for his guidance that shaped the entire direction of this work.

"When I let go of what I am, I become what I might be. When I let go of what I have, I receive what I need."

John Heider, The Tao of Leadership (1985)

Contents

Abstract **i**

1 Introduction **1**
1.1 Context . . . 1
1.2 Problem Statement . . . 2
1.3 Objectives and Research Questions . . . 3
1.4 Research Methodology . . . 4
1.5 Thesis Structure . . . 4

2 Background and Related Work **7**
2.1 ETO Manufacturing and Decision-Making . . . 7
2.2 Machine Learning for Manufacturing Duration Prediction . . . 8
2.3 Supply Chain Risk and Commodity Forecasting . . . 9
2.4 LLM-Based Agents for Decision Support . . . 10
2.5 Summary . . . 11

3 Research Framework and System Requirements **13**
3.1 Industrial Context . . . 13
3.2 Decision Making Challenges . . . 14
3.3 Scope of the Proposed System . . . 14
3.4 Design Requirements . . . 15
3.5 Problem Decomposition . . . 15
3.5.1 Commodity Risk Forecasting . . . 15
3.5.2 Manufacturing Duration Prediction and Project Planning . . . 16
3.5.3 Agent-Based Integration . . . 16
3.6 Evaluation Framework . . . 16

4 Design and Development of the Decision Support System **17**
4.1 Overall System Architecture . . . 17
4.2 Commodity Risk Forecasting Module . . . 18
4.3 Manufacturing Duration Prediction Module . . . 24
4.4 Project Planning Module . . . 29
4.5 LLM-Based Decision Support Agent . . . 30

5 Evaluation and Discussion **33**
5.1 Evaluation of Commodity Risk Forecasting . . . 33
5.2 Evaluation of Manufacturing Duration Prediction . . . 38
5.3 Evaluation of the LLM-Based Agent . . . 42
5.4 Discussion of RQ1: Uncertainty in ETO Decision Processes . . . 44

5.5 Discussion of RQ2: Integrating Heterogeneous Predictive Tools into an LLM-Based Agent . . . 46
5.6 Managerial Implications . . . 47

6 Conclusions **49**
6.1 Summary of Findings . . . 49
6.2 Answers to the Research Questions . . . 50
6.3 Contributions and Limitations . . . 51
6.4 Future Research . . . 52

References **55**

A Manufacturing-Order Dataset Construction **59**
A.1 Data-Quality Exclusion Rules . . . 59
A.2 Scope Restriction and the Excluded Tail . . . 60
A.3 Modelling-Population Distribution . . . 61

B Commodity Disruption Forecasting: Supplementary Materials **63**
B.1 Alert Policy, Utility Function and Regression Sample Weighting . . . 63
B.2 Latent-Regime Feature Construction . . . 65
B.3 Detailed Evaluation Design and Metric Definitions . . . 67
B.4 Alert-Utility Robustness Checks . . . 68
B.5 Stricter Fixed-Policy Re-Score . . . 69
B.6 Classification and Calibration Diagnostics . . . 70
B.7 Latent Regime Diagnostics . . . 72
B.8 Regime Block Sensitivity Analysis . . . 75
B.9 Regression-Layer Ablation Details . . . 77
B.10 Classifier Architecture Ablation . . . 78
B.11 SHAP Attribution Analysis . . . 79

C Additional Manufacturing-Order Evaluation Diagnostics **81**
C.1 First-Stage Classifier Diagnostics . . . 81
C.2 Residual and Subgroup Diagnostics . . . 82
C.3 Hyperparameter Search and Selection . . . 85
C.4 Feature-Group Ablation . . . 86
C.5 Duration Prediction Interval Diagnostics . . . 87
C.6 Comparison of Prediction-Interval Methods . . . 88
C.7 Temporal Stability . . . 90

D Shop-Floor Simulation: Implementation Details **91**
D.1 Operation-Time Representation . . . 91
D.2 Daily Advancement Mechanics . . . 91
D.3 Synthetic Arrivals and Material Eligibility . . . 92
D.4 Capacity Banding and Candidate Scoring . . . 93
D.5 Throughput Dynamics Validation . . . 94

E Agent Layer: Routing Validation and Interface **97**
E.1 Routing and Grounding Validation . . . 97
E.2 Agent Interface Walkthrough . . . 98

Acronyms

AI artificial intelligence.

API application programming interface.

CI confidence interval.

ECE expected calibration error.

EM expectation-maximisation.

ERP enterprise resource planning.

ETO engineer-to-order.

FRED Federal Reserve Economic Data.

FX foreign exchange.

GDELT Global Database of Events, Language, and Tone.

GKG Global Knowledge Graph.

GMM Gaussian mixture model.

LLM large language model.

LME London Metal Exchange.

LSTM long short-term memory.

MAE mean absolute error.

ML machine learning.

OOF out-of-fold.

PR-AUC precision-recall area under the curve.

RFQ request for quotation.

rMAE relative mean absolute error.

ROC-AUC receiver operating characteristic area under the curve.

SHAP SHapley Additive exPlanations.

XGBoost Extreme Gradient Boosting.

List of Figures

4.1 Overall architecture of the decision-support system 18

5.1 Mean absolute SHAP contribution by feature group, expressed as a percentage of the total, for the regressor evaluated on the test set. Groups are sorted in descending order of contribution. 42

B.1 OOF alert episodes for copper (top) and nickel (bottom) overlaid on the weekly price series. Downside windows are shaded red; upside windows are shaded blue. Disruption threshold levels $\pm\tau$ are indicated. 71
B.2 Reliability diagrams for the copper and nickel directional disruption classifiers (OOF). Left: copper. Right: nickel. Observed disruption frequency is plotted against mean predicted probability within equal-width bins. The diagonal indicates perfect calibration; bar widths are proportional to bin population. 72
B.3 Temporal coherence diagnostics for the fold-replayed latent regime assignments. Left: copper. Right: nickel. The plots show whether inferred dominant states persist in coherent blocks over time rather than flipping week to week. 74
B.4 Empirical transition matrices for the fold-replayed latent regime assignments. Left: copper. Right: nickel. Diagonal mass reflects state persistence, while off-diagonal mass shows that transitions remain possible and the regime process is not degenerate. 74
B.5 Mean absolute SHAP contribution by feature family for the commodity classifiers (OOF), expressed as a percentage of each target's total. Top row: copper downside and copper upside. Bottom row: nickel downside and nickel upside. 80

C.1 Precision-recall curve for the first-stage classifier on the test set. The dashed horizontal line marks the prevalence of multi-day orders (0.595). The operating point at threshold 0.284 is marked. PR-AUC = 0.977. 82
C.2 Predicted versus actual duration for the full pipeline on the test set ($n = 3{,}249$). Points on the diagonal indicate perfect prediction. The dashed lines bound the $\pm 15\%$ within-band region. Colour intensity indicates point density. 82
C.3 Distribution of signed residuals (actual $-$ predicted, working days) for the planned-hours-derived estimate (left, gray fill) and pipeline (right, coloured fill) within each duration bracket. The horizontal line at zero marks unbiased prediction. Bracket sample sizes are as reported in Table 5.6. 83
C.4 Relative MAE (%) by workstation type for the pipeline (coloured bars) and the planned-hours-derived estimate (gray bars). The vertical dashed line marks the overall pipeline rMAE%. 84
C.5 Relative MAE (%) by project type for the pipeline (coloured bars) and the planned-hours-derived estimate (gray bars). The overall pipeline rMAE% is indicated by the dashed line. 85

E.1 Single manufacturing-order query. The agent returns the point duration estimate with its P10–P90 uncertainty interval and a natural-language breakdown of the dominant SHAP drivers, translating feature attributions into plain language rather than presenting raw importance scores. 99
E.2 Project-planning query. The agent invokes the planning tool and reports the recommended start sequence and expected finish for the project's manufacturing orders, together with the constraint diagnostics surfaced from the simulation backend. 100
E.3 Commodity disruption query. The agent reports the current alert status and renders the price-overlay chart produced by the commodity backend, illustrating how tool-generated visual artifacts are surfaced alongside the textual response. 101

List of Tables

5.1 Alert utility and episode-level statistics for the OOF evaluation period. U_d is in percentage points of price return. Directional accuracy is the fraction of episodes in which the price moved in the alerted direction by at least 5% at some point during the active window. $\bar{V}_e^+$ is the mean gated correct-direction variation per episode. 34

5.2 Commodity classifier and alert-policy ablation results. Higher PR-AUC and higher U_{total} are better. ΔU is measured relative to each metal's baseline alert policy. The two regressor-only ablations that remove the disruption-focused sample weighting or the classifier-probability feature leave these quantities unchanged and are reported separately in Table B.12. 35

5.3 Commodity model-versus-baseline comparison. PR-AUC baseline is the no-skill prevalence baseline for each task. Alert-utility baseline is the frequency-matched momentum policy described above. Higher values are better for both metrics. . . 36

5.4 Commodity regression performance on the two evaluation populations. *Alert-firing* comprises all OOF weeks where the classifier fires an alert, including re-fires; n_{ep} is the number of distinct episodes represented. *True disruption* comprises all OOF weeks where a disruption did take place, independent of the alert policy. Positive Bias indicates systematic under-prediction of move magnitude. Coverage is the fraction of weeks where $r_{\text{true}} \in$ [ci_lower, ci_upper]; the target is 0.80. Width is the mean interval width in percentage return units. 37

5.5 Overall performance on the test set ($n = 3{,}249$ manufacturing orders). InBand is the fraction of predictions within $\pm 15\%$ of actual duration. W1d is the fraction of predictions with absolute error ≤ 1 working day. rMAE% is relative MAE normalised by the training-group median duration. Bias is the mean signed error (actual $-$ predicted); positive values indicate systematic under-prediction. Ridge is a regularised linear regression and Single regressor a single XGBoost regressor trained on all orders without classifier routing; both share the pipeline's feature set and isolate, respectively, the model-class and two-stage-architecture decisions. . . 39

5.6 Regressor performance on multi-day orders segmented by actual duration bracket ($n = 1{,}932$ orders with a multi-day actual duration). 40

5.7 Informal expert evaluation scores (1–5) by dimension. Each column is one evaluator; the final column reports the per-dimension mean. 43

A.1 Manufacturing orders flagged by each data-quality rule on the full study population. A single order may trigger several rules, so the counts are not mutually exclusive (417 orders are flagged by two or more); 3,478 distinct orders are removed from the modelling population in total. 60

A.2 Distribution of the cleaned orders excluded by the 22-working-day scope restriction ($n = 429$, 2.57% of the cleaned population). 60
A.3 Summary statistics of the target variable (effective duration in working days) on the modelling population ($n = 16{,}245$). 61
A.4 Composition of the modelling population by realised duration bracket ($n = 16{,}245$). 61
A.5 Structural complexity of the modelling population: per-order distribution of operation, workstation, and component counts ($n = 16{,}245$). 61

B.1 Leave-one-episode-out robustness of reference alert utility. U_{total} is reported in percentage points of price return. 68
B.2 Cost-sensitivity range for reference alert utility. The grid contains 18 settings per metal, including the baseline scoring rule and one-at-a-time or joint cost multipliers. 69
B.3 Strict ex-post re-score of the reference alert policies. Both utilities are in percentage points of price return and are computed on the same fixed alert episodes. . . . 69
B.4 Strict ex-post re-score of commodity ablation totals. ΔU^{strict} is relative to the strict re-score of each metal's full reference model. 70
B.5 OOF classification performance for the copper and nickel directional disruption classifiers. Precision, recall, and F_1 are evaluated at the operating thresholds selected by the alert utility optimisation (Section 4.2). 70
B.6 Classifier calibration metrics on the full OOF sample. The reference Brier is the prevalence-only value $p(1-p)$, i.e. the score obtained by always predicting the base rate; a model Brier below this reference ($\Delta < 0$) indicates probabilistic information beyond the base rate. ECE is the expected calibration error over ten equal-width probability bins. 71
B.7 Empirical replayed profile of the copper latent regimes. Cluster weight is the share of weeks assigned to each dominant regime in the replayed evaluation sample. Forward return is the mean 8-week ahead return conditional on the dominant regime. 72
B.8 Empirical replayed profile of the nickel latent regimes. Cluster weight is the share of weeks assigned to each dominant regime in the replayed evaluation sample. Forward return is the mean 8-week ahead return conditional on the dominant regime. 73
B.9 Marginal sweep over the post-transition anchor year, with $T = 1.8$ and $\alpha = 0.10$ fixed at the baseline. *Fit rows* is the number of weekly observations used to fit the GMM. *Pre-2019 conc.* is the fraction of weeks in $[2015, \text{anchor})$ assigned to a single dominant regime; the health gate requires this value to be at most 0.90. *Valid* indicates whether all four health gates are satisfied. PR-AUC values are produced by walk-forward refit/score, pooled across folds, and directly comparable to the production pipeline's headline regime-only figures. 76
B.10 Marginal sweep over the posterior-softening temperature T, with the anchor year fixed at 2019 and $\alpha = 0.10$. Columns as in Table B.9. 76
B.11 Marginal sweep over the prior-blending coefficient α, with the anchor year fixed at 2019 and $T = 1.8$. Columns as in Table B.9. 77
B.12 Regressor-only ablations on the alert-firing population. Both variants leave classifier discrimination and alert utility identical to the baseline (cf. Table 5.2) and act solely on the magnitude estimates. One variant removes the regressor's disruption-focused sample weighting, and the other removes the classifier-probability feature supplied to the regressor. mean absolute error (MAE) is in percentage return units; positive Bias indicates under-prediction. 78

B.13 Nickel classifier architecture ablation. Alert utility is the total over both directions (basis points of captured directional variation, net of penalties); PR-AUC is reported per direction. Best per column in bold. 78
B.14 SHAP attribution by feature family for the copper disruption classifiers (OOF), as a percentage of total mean absolute SHAP contribution. Families are listed in descending order of combined attribution across both directions. 79
B.15 SHAP attribution by feature family for the nickel disruption classifiers (OOF), as a percentage of total mean absolute SHAP contribution. Families are listed in descending order of combined attribution across both directions. 79

C.1 First-stage classifier performance on the test set ($n = 3{,}249$). $F_{\beta=1.5}$ weights recall more heavily than precision, reflecting the asymmetric cost of misclassifying a multi-day order as same-day. 81
C.2 Pipeline performance by primary workstation type on the test set. For orders routed through multiple workstation types (*Mixed*), the multi-type routing itself is treated as the workstation segment. rMAE% is normalised by the training-group median for each category. The Automation segment ($n = 13$) is too small for reliable inference and is included for completeness only. 83
C.3 Pipeline performance by project type on the test set. Project types are ordered by descending sample size. 85
C.4 Hyperparameter search grid and selected values for the manufacturing-order duration models. The same grid is searched for all three models by five-fold cross-validation against the per-model criterion below; the number of boosting rounds is set by early stopping (patience 50, ceiling 20,000) rather than grid-searched. . . . 86
C.5 Manufacturing-order duration feature-group ablation on the held-out test set. Each row removes one feature group and retrains the full two-stage pipeline. #grp is the number of features in the respective group; ΔMAE and ΔR^2 are relative to the all-features baseline; ΔMAE/ft normalises the effect by group size. Positive ΔMAE / negative ΔR^2 means the group earns its place. 87
C.6 Overall P10–P90 interval diagnostics for regressor-routed manufacturing orders, excluding regressor-routed cases with realised zero-day duration. 88
C.7 P10–P90 interval diagnostics by realised duration bracket, excluding regressor-routed cases with realised zero-day duration. 88
C.8 Temporal stability of duration-interval diagnostics after excluding regressor-routed cases with realised zero-day duration. 88
C.9 Overall interval-method comparison on the regressor-routed test population ($n = 1{,}865$). Coverage targets 80%. <P10 and >P90 are the fractions of realised durations below the lower and above the upper bound; W. mean and W. med. are the mean and median interval width in working days. Lower pinball loss is better. . . 89
C.10 Conditional P10–P90 coverage (%) by realised duration bracket for each interval method, on the same population as Table C.9. The nominal target is 80% within every bracket. 89
C.11 Pipeline performance on the first and second chronological halves of the test set. The split date falls on 14 October 2025. 90

D.1 Cumulative drained-throughput validation 95
D.2 Factory-day validation after summing all workstations. This table measures day-by-day timing accuracy rather than cumulative horizon follow-through. 95

D.3 Factory-level backlog state accuracy at the start and end of each simulated horizon. Workstations are summed before computing the metrics. 96

E.1 Scenario coverage of the scripted agent routing and grounding validation suite. . 97
E.2 Results of the scripted agent routing and grounding validation suite. 98

Chapter 1

Introduction

1.1 Context

Engineer-to-order (ETO) manufacturing is increasingly relevant in industries where customers demand tailored solutions rather than standard products. In ETO contexts, each project is partially or fully unique, design and engineering activities are significant, and the quoting phase often occurs before detailed specifications are available (Gosling & Naim, 2009). This structural uncertainty permeates the entire decision-making chain: from early commercial bidding through production scheduling to procurement and resource allocation, managers are routinely asked to commit to consequential decisions with incomplete, ambiguous, or unevenly distributed information (Burggräf et al., 2020).

This informational opacity manifests in several ways. Internally, the durations of manufacturing orders are difficult to estimate reliably because the scope of the project varies widely, historical records are often inconsistent, and estimates are highly dependent on expert judgement, introducing subjectivity that varies between estimators and over time (Burggräf et al., 2020). Inaccurate duration estimates propagate downstream: they distort capacity planning, misalign procurement schedules, and compromise bid pricing, leading either to underquoting, which erodes margin, or overquoting, which reduces competitiveness (Nederkoorn, 2017).

Externally, ETO manufacturers are exposed to supply chain volatility, particularly for critical inputs such as industrial metals, where price signals and disruption risk are difficult to interpret and act on without dedicated analytical infrastructure (Baryannis et al., 2019).

Recent progress in data availability and machine learning (ML) has motivated the development of predictive systems that can partially compensate for these information gaps. Compared to purely experience-based approaches, ML models can learn patterns from historical records and provide estimates that are faster, more consistent, and less dependent on individual expertise (Caputo & Pelagagge, 2008; Elmousalami, 2021).

However, deploying ML in ETO settings remains challenging: models are trained in isolation, their outputs are not always interpretable to decision-makers, and no single model captures the full complexity of operational decision-making. This work proposes a different architectural

perspective: rather than treating individual predictive models as standalone tools, we frame the decision-support problem as one of agent-mediated reasoning across multiple ML systems.

Specifically, we present an artificial intelligence (AI) agent architecture that integrates and interprets the outputs of heterogeneous predictive models, including a manufacturing order duration estimator and a commodity supply risk monitor, providing decision-makers with a unified and interpretable interface to query and reason about uncertainty across operational and procurement domains. The agent acts as an analytical intermediary: it does not replace the underlying models but gives them a shared reasoning layer through which a decision-maker can ask complex, context-sensitive questions and receive grounded, transparent answers (Qu et al., 2024).

This framing directly addresses a key limitation of current ML use in ETO contexts: the gap between model availability and operational usefulness. A predictive model that a manager cannot interrogate, contextualise, or trust will not change decisions. By combining structured ML outputs with an agent capable of multi-model reasoning, this thesis seeks to make uncertainty in ETO environments more legible and therefore more manageable (Simchi-Levi et al., 2025).

In practice, however, prediction alone is not sufficient. Estimating how long an order is likely to take only solves part of the planning problem; managers must also decide when orders should start, taking into account current congestion, routing dependencies, component availability, and competition for capacity across projects. This creates a need not only for predictive models, but also for planning-oriented tools that can translate those predictions into operational decisions.

This work is also aligned with the United Nations Sustainable Development Goals, particularly SDG 8, *Decent Work and Economic Growth*, and SDG 9, *Industry, Innovation and Infrastructure* (United Nations General Assembly, 2015). In relation to SDG 8, the thesis addresses productivity and operational efficiency in an industrial setting by supporting more reliable planning, reducing dependence on informal estimation practices, and improving the consistency of decision-making under uncertainty. In relation to SDG 9, the work contributes to industrial innovation by developing a data-driven decision-support architecture that integrates predictive models, simulation, and an agent based on a large language model (LLM) within an engineer-to-order manufacturing context. The contribution is therefore not a direct policy intervention, but an applied industrial system that supports more resilient, transparent, and innovation-oriented manufacturing decision processes.

1.2 Problem Statement

As outlined above, ETO managers are regularly required to make high-stakes planning and procurement decisions under conditions of pronounced uncertainty regarding manufacturing lead times and supply chain risks. Although predictive models for lead time estimation and supply chain risk assessment have shown promise, they are typically developed as isolated tools, with heterogeneous inputs, outputs, and assumptions (Baryannis et al., 2019; Burggräf et al., 2020). As a result, decision-makers struggle to interrogate, compare, and contextualise model outputs at the level of individual projects, and the gap between model availability and operational usefulness

persists. The core problem addressed in this thesis is how to provide ETO decision-makers with a unified, interpretable interface that can mediate between multiple predictive models and support reasoning about cross-cutting uncertainties in order duration and planning as well as commodity supply risk.

1.3 Objectives and Research Questions

The central challenge addressed in this thesis is to connect predictive models with the practical decisions faced by planners and buyers in ETO environments. In such settings, decision support must account for noisy data, heterogeneous prediction targets, deployment constraints, and the need to communicate uncertainty at the level of specific orders, projects, and supply risks, rather than only through aggregate performance metrics.

The overall goal of this thesis is to design, develop, and evaluate an AI-based decision-support agent for ETO environments. The proposed system integrates predictive and planning components for manufacturing order duration estimation, project planning, and commodity-related supply risk monitoring within a unified LLM-orchestrated interface.

Concretely, the thesis pursues the following objectives:

- To characterise the decision-making challenges associated with order duration uncertainty, project planning, and commodity-related supply risks in an industrial ETO context.
- To develop predictive and planning components that support these decision areas using company data and deployment-oriented interfaces.
- To design an LLM-based agent architecture capable of orchestrating these heterogeneous tools and communicating their outputs in a coherent and interpretable way.
- To evaluate the predictive components in terms of performance, calibration, robustness, and practical alert utility, and to assess the integrated agent through automated tests and feedback from its potential users.

These objectives give rise to the following research questions, informed by recent work on tool-using LLM agents (Qu et al., 2024) and LLM-based operations decision support (Simchi-Levi et al., 2025):

RQ1: How do order duration uncertainty, project planning needs, and commodity-related supply risks manifest in ETO decision processes, and what limitations do current predictive approaches exhibit in practice?

RQ2: How can heterogeneous predictive and planning tools be integrated into an LLM-based decision-support agent that communicates model outputs coherently and interpretably for ETO decision-makers?

This thesis makes three main contributions:

- **An agent-based architecture for ETO decision support.** The thesis proposes an LLM-orchestrated architecture for integrating heterogeneous predictive and planning services into a unified decision-support interface.

- **A prototype system grounded in industrial data.** The thesis develops and connects components for manufacturing order duration estimation, simulation-based project planning, and commodity supply risk monitoring using data and workflows from an industrial ETO context.

- **Empirical findings and design implications.** The thesis evaluates the predictive components in terms of accuracy, calibration, temporal robustness, directional classification performance, and alert utility, and discusses the integrated agent based on automated testing, iterative development, user feedback, and practical use.

1.4 Research Methodology

Methodologically, the thesis follows a design-and-evaluation approach that combines predictive modelling, simulation-based decision support, and agent-oriented system integration. The work is grounded in industrial data from an ETO context, complemented by external market, macroeconomic, and news-derived data for the commodity risk component.

Three main technical components are developed. First, a manufacturing order duration estimator is trained on historical enterprise resource planning (ERP) records to predict order-level processing times and associated uncertainty. Second, a project planning tool uses these duration estimates within a shop-floor simulation framework to support start-date and delivery-risk analysis. Third, a nickel/copper disruption forecasting pipeline is built from market, macroeconomic, and news-derived signals to monitor commodity-related supply risks.

These components are integrated through an LLM-based agent architecture that can query the different tools, coordinate their outputs, and communicate the resulting analysis through a grounded conversational interface. The predictive components are evaluated quantitatively in terms of accuracy, calibration, robustness, and alert utility, while the planning component is assessed through scenario-based analysis. The agent layer is evaluated through automated tests of tool exposure, routing, and orchestration behaviour, and is further discussed on the basis of iterative development, informal feedback from managers, and the author's own experience of using the system.

1.5 Thesis Structure

The remainder of this thesis is organised as follows. Chapter 2, *Background and Related Work*, surveys the literature on ETO manufacturing, predictive modelling for manufacturing times and supply risks, and LLM agents that use external tools. Chapter 3, *Research Framework and System*

Requirements, translates that literature into the thesis-specific decision context, scope, requirements, and evaluation logic.

Chapter 4, *Design and Development of the Decision Support System*, presents the data sources, feature engineering, predictive pipelines, simulation logic, and the LLM-based agent architecture that orchestrates these models. Chapter 5, *Evaluation and Discussion*, reports and interprets the quantitative performance of the predictive models together with the qualitative assessment of the agent layer.

Finally, Chapter 6, *Conclusions*, returns to the goals established in this introduction, summarises the main findings and contributions, reflects on limitations, and outlines future developments.

Chapter 2

Background and Related Work

This chapter surveys the scientific and technical literature relevant to decision support in ETO environments under uncertainty about manufacturing order durations and commodity-related supply risks. It first introduces the characteristics of ETO environments and the associated decision-making challenges. It then surveys predictive approaches for estimating manufacturing processing times, as well as work on supply chain risk prediction and commodity price forecasting. Finally, it reviews recent advances in LLMs that can use external tools and act as decision-support agents. The chapter concludes by synthesising the research gap that motivates the thesis.

2.1 ETO Manufacturing and Decision-Making

ETO manufacturing refers to production environments in which each customer order is, to a significant extent, engineered and configured to specific customer requirements rather than drawn from a predefined catalogue of standard products (Gosling & Naim, 2009). Projects are often one-of-a-kind or produced in very small batches, with substantial design and engineering effort preceding or overlapping with production. This contrasts with make-to-stock and assemble-to-order contexts, where product structure and routings are largely fixed, and historical data can be reused more directly.

The ETO mode of operation creates distinctive planning and control challenges. Customer requirements may change during the project, engineering changes can propagate downstream into manufacturing and procurement, and the scope and complexity of work are often only partially known at the quoting stage. As a result, managers must make acceptance, pricing, capacity planning, and procurement decisions based on incomplete and evolving information (Burggräf et al., 2020). Manufacturing times are not merely a function of routings and standard processing times, but depend on engineering effort, supplier responsiveness, and interactions between parallel projects.

Gosling and Naim (2009) and Burggräf et al. (2020) analyse ETO processes and decision points in detail, identifying recurring themes such as the importance of early delivery time promises,

the need to balance engineering and production resources across projects, and the impact of variability on schedule adherence. These studies consistently highlight uncertainty in order durations and supply risks as central obstacles to effective decision-making (Burggräf et al., 2020; Gosling & Naim, 2009).

2.2 Machine Learning for Manufacturing Duration Prediction

In response to these challenges, a growing literature has focused on predicting manufacturing times and delivery dates in ETO settings. Burggräf et al. provide a systematic review of approaches for manufacturing time prediction in ETO environments, identifying a range of methods from analytical queuing models and regression-based techniques to more recent ML approaches (Burggräf et al., 2020). Their review highlights that data availability is often limited, feature sets are heterogeneous, and evaluation procedures are not standardised across studies.

Case-based and ML-based approaches have been proposed to estimate delivery times already at the request for quotation (RFQ) stage, using historical orders and project characteristics as predictors. For example, some case studies demonstrate that supervised learning models trained on past projects can reduce delivery time estimation errors compared to expert judgement alone, particularly when capturing interactions between engineering complexity, routing choices, and workload levels (Rokoss et al., 2024). However, these models are typically embedded in local decision-support tools and are rarely integrated across the broader planning and procurement processes.

Related evidence from high-mix, low-volume job shops points in the same direction, showing that data-driven models can benefit substantially from contextual shop-floor variables such as workload and machine state (Müller & Grumbach, 2023).

Beyond ETO and job-shop studies specifically, the prediction of activity and case durations has been examined extensively under the banner of predictive process monitoring, where supervised and sequence models estimate the remaining time of in-progress process instances. A cross-benchmark survey by Verenich et al. (2019) finds that tree-based and gradient-boosting models are consistently competitive with more complex sequential architectures for remaining-time regression, which motivates the gradient-boosted approach adopted in this thesis.

Despite these advances, several limitations remain. First, most models are developed and evaluated on data from a single facility or product family, with limited evidence of generalisability. Second, interpretability is often limited to feature importance scores or partial dependence plots; even unified attribution methods such as SHapley Additive exPlanations (SHAP) (Lundberg & Lee, 2017) are usually surfaced as raw numerical scores, which may not be sufficient for managers who need to understand and justify delivery promises on a case-by-case basis. Third, the models are usually used in isolation: they produce estimates that must be manually reconciled with other sources of uncertainty, such as supplier performance or commodity price volatility, in the minds of decision-makers.

Summary and Limitations

Synthesising the literature on ETO and job shop manufacturing time prediction, several patterns emerge. There is clear evidence that ML-based approaches can reduce estimation errors compared to purely experience-based methods, particularly when they exploit historical order data and contextual information (Burggräf et al., 2020; Rokoss et al., 2024). At the same time, prevailing approaches are fragmented, focus on single prediction targets, and provide limited support for integrated reasoning about uncertainty across multiple models and decision domains. This fragmentation motivates the search for architectures that can both integrate heterogeneous predictive models and expose their outputs in a way that is more accessible to decision-makers.

2.3 Supply Chain Risk and Commodity Forecasting

This section reviews work relevant to the commodity disruption component of the thesis. The focus is on data-driven approaches that use market, macroeconomic, and textual signals to anticipate commodity-related disruptions relevant to procurement decisions.

Data-Driven Risk Monitoring

Beyond internal manufacturing processes, ETO firms are exposed to external risks arising from their supply chains, including delivery delays, supplier failures, and disruptions in the availability or price of critical inputs. ML-based approaches to supply chain risk prediction seek to anticipate such events by learning from historical data on deliveries, supplier performance, and external signals. Baryannis et al. propose a framework for predicting supply chain risks using ML, explicitly analysing the trade-off between predictive performance and model interpretability (Baryannis et al., 2019). Their study shows that complex models can achieve higher accuracy but are often harder to explain, while simpler models may be more transparent but less accurate.

Other works extend this line of research by incorporating additional data sources such as news, social media, and macroeconomic indicators, or by focusing on specific risk types such as port congestion or transport disruptions. Across these studies, ML is primarily viewed as a component within broader risk management systems, often providing risk scores or alerts that must be interpreted and acted upon by human planners.

Commodity and Metals Price Forecasting

In parallel, there is a substantial literature on forecasting commodity prices, including industrial metals such as nickel and copper, which are critical inputs for many ETO products. Time-series and ML models have been applied to forecast metal prices across different horizons, using techniques such as autoregressive models, support vector regression, long short-term memory (LSTM) networks, and hybrid decompositional approaches (Liu et al., 2020).

A well-documented pitfall in forecasting volatile financial and commodity prices is to assess complex models only relative to each other, without benchmarking against simple naive baselines

such as a no-change forecast. Beck et al. show that for a large collection of noisy financial and macroeconomic time series, many machine learning studies report gains only relative to other sophisticated methods, even though none of the models consistently outperforms a naive persistence benchmark, making their accuracy comparisons hard to interpret in absolute terms (Beck et al., 2025). This issue is particularly salient at very short horizons: a model that predicts next week's nickel or copper price by closely tracking last week's level can obtain apparently low errors on volatile data, while providing little directional information or operational value for planning decisions. The literature therefore points to the importance of aligning the prediction target and the evaluation horizon with the operational decision that the forecast is meant to support.

In commodity procurement under price uncertainty, data-driven policies that use real-time feature information have been shown to outperform internal benchmark practices and generate substantial cost savings (Mandl & Minner, 2023). Comparable results have been reported in industrial raw material procurement settings, where ML-based price forecasting coupled with an optimised purchasing policy yields measurable reductions in total procurement cost (Lee et al., 2022).

Summary and Limitations

The literature on supply chain risk prediction and commodity price forecasting demonstrates that ML can provide nontrivial predictive power for external risk factors relevant to ETO firms (Baryannis et al., 2019). Yet, as with manufacturing time prediction, these models tend to exist as separate tools, with their own interfaces, metrics, and assumptions. There is little work on how to combine internal predictive models (e.g., order durations) and external ones (e.g., commodity price forecasts) into a coherent support system where a manager can transparently reason about the joint impact of multiple uncertainties on project outcomes.

2.4 LLM-Based Agents for Decision Support

This section reviews the literature relevant to the agent layer of the thesis. It first considers work on tool-using LLMs and agent architectures, then turns to applications of LLMs in supply chain and operations decision support, where the main concern is not only model capability but also grounding, interpretability, and organisational usefulness.

LLM Agents and Tool Learning

Recent advances in LLMs have enabled models that not only generate text but also use external tools, call application programming interfaces (APIs), and interact with software systems in an agentic manner. Early method papers such as ReAct (Yao et al., 2023) and Toolformer (Schick et al., 2023) demonstrate that LLMs can be guided to interleave natural-language reasoning with tool calls, enabling them to retrieve information, perform calculations, or interact with environments while maintaining a coherent chain of thought. Building on these ideas, survey articles on tool learning with LLMs and on LLM agents provide systematic overviews of techniques for

representing tools, prompting models to select and sequence tool calls, and training or finetuning models to improve tool use and reduce hallucinations (Luo et al., 2025; Qu et al., 2024).

These surveys identify a common pattern: LLMs can serve as a flexible orchestration layer that decides when and how to invoke external tools, integrates their outputs into subsequent reasoning, and communicates results to users in natural language. This pattern is attractive for domains where multiple specialised models or services already exist, but are difficult for end users to access or combine.

LLMs in Supply Chain and Operations Decision Support

A growing line of work explores the use of LLMs as decision-support interfaces in operations and supply chain management. Simchi-Levi et al. discuss how LLMs can democratise access to existing optimisation tools by allowing planners to pose questions and what-if scenarios in natural language, while the LLM translates these into appropriate calls to underlying solvers and data services (Simchi-Levi et al., 2025). Their perspective positions LLMs as an orchestration and explanation layer on top of established analytics and optimisation components, rather than as autonomous decision-makers.

Other contributions, including theses and conceptual papers, investigate the potential of generative AI and LLMs in supply chain management more broadly, considering use cases such as demand forecasting explanation, supplier risk analysis, and interactive scenario planning (Dhara & Delgado Barba, 2024). These works highlight both opportunities (improved accessibility, faster information retrieval) and challenges (hallucinations, data governance, robustness) associated with deploying LLM-based systems in operational contexts.

Summary and Limitations

The literature on LLM-based tool-using agents suggests that LLMs are well-suited to acting as intermediaries between human decision-makers and heterogeneous computational tools, including ML models and optimisation solvers (Luo et al., 2025; Qu et al., 2024; Simchi-Levi et al., 2025). However, most existing applications focus on generic information-seeking tasks, software engineering, or high-level supply chain planning. There is limited work on applying LLM agents to ETO manufacturing, and the literature surveyed here provides little guidance on how manufacturing order duration prediction, project-level planning, and commodity-related supply risk analysis might be integrated into a unified, interrogable interface for project-level decision-making.

2.5 Summary

This chapter has surveyed prior work in four main areas: ETO manufacturing and its decision-making challenges; ML-based prediction of manufacturing order duration; ML-based approaches

to commodity-related supply risk and commodity price prediction; and emerging work on LLM-based tool-using agents and decision-support systems. Across these domains, there is strong evidence that ML can improve the prediction of key variables such as order durations and commodity prices, and that LLMs can serve as powerful interfaces to complex analytical tools (Baryannis et al., 2019; Burggräf et al., 2020; Qu et al., 2024; Simchi-Levi et al., 2025).

At the same time, several gaps remain. Predictive models in ETO contexts are often developed and deployed in isolation, focusing on single targets and offering limited support for integrated reasoning about cross-cutting uncertainties. Supply chain risk and commodity price models are typically embedded in separate analytics environments, disconnected from the tools used by production planners and project managers. Existing work on LLM agents and tool use provides general architectures for orchestrating external tools, but offers little guidance on how to tailor these architectures to ETO decision processes or how to connect them to domain-specific predictive models in a principled way.

These gaps motivate a thesis-specific problem formulation that bridges the literature and the implemented system. The next chapter therefore defines the operational decision context, the scope of the proposed support system, and the requirements that guide the methodological choices developed later in the thesis.

Chapter 3

Research Framework and System Requirements

This chapter translates the literature gaps identified in the previous chapter into the specific decision-support problem addressed in the thesis. It first describes the industrial decision context, then decomposes the problem into manufacturing duration prediction, commodity disruption forecasting, and LLM-based tool orchestration, before defining the evaluation perspective used for the proposed system.

3.1 Industrial Context

The system developed in this thesis is grounded in an industrial ETO manufacturing context. The case company designs and produces customer-specific industrial solutions for external clients, with many projects engineered substantially from scratch in response to customer specifications and others following more standard approaches for other clients. Production is organised around a mixed-workstation shop floor, where manufacturing orders progress through variable sequences of operations including cutting, forming, machining, welding, assembly, and finishing, with routings that differ considerably across project types and complexity levels. Orders range from single-operation jobs completable within a working day to multi-stage assemblies spanning several days and requiring coordinated procurement of critical materials.

Within this environment, decision-makers must act before the full operational picture is known. At order and project launch, planners need to estimate manufacturing durations that feed directly into capacity allocation, delivery promises, and procurement timing. At the same time, procurement decisions for metals such as nickel and copper must often be taken weeks in advance under volatile market conditions. These decisions are operationally linked: uncertain production timing changes when materials are needed, while uncertain material conditions affect when and how procurement should occur.

3.2 Decision Making Challenges

The central problem addressed in this thesis is how to support ETO decision-makers who must reason simultaneously about internal uncertainty in manufacturing order duration and project planning as well as external uncertainty in commodity-related supply risk. At the case company, each of these tasks is currently handled informally and in isolation. Commodity exposure for metals such as nickel and copper is assessed essentially by hand: buyers and their managers track prices and decide on purchase timing based on their own reading of the market, without any systematic forecasting support.

Manufacturing order durations are not estimated by any dedicated model either; they come either from the personal experience of those launching the orders or from engineers' figures that are frequently inaccurate. The only signal on current shop-floor load is a coarse load proxy in the ERP, the summed planned hours of all concurrently open orders, and it is not systematically combined with these estimates. These duration estimates are then handed off to yet other people who carry out the actual project planning. Predictive tooling that could address parts of this problem exists in the literature, but it is typically isolated: different tools use different inputs, produce different outputs, and require users to interpret and reconcile their results manually.

In practice, this fragmentation means that even where estimates are available they do not automatically translate into better decisions. A planner may receive a duration figure of uncertain provenance, and a buyer may form a private view of where metal prices are heading, yet neither has an integrated picture of how these uncertainties interact at project level, nor a shared, reproducible basis for the numbers they act on. The problem is therefore not only predictive, but also integrative and communicative: the system must replace ad-hoc, experience-based judgements with consistent model outputs, combine those heterogeneous outputs, and present them in a form that supports operational reasoning.

3.3 Scope of the Proposed System

This thesis addresses that problem through a decision-support prototype with four tightly connected components: three predictive and planning services and an LLM-based agent layer that oversees and orchestrates them:

- a predictive pipeline for nickel and copper disruption forecasting over a medium-term procurement horizon;
- a predictive pipeline for manufacturing order duration estimation;
- a project planning tool that simulates shop-floor dynamics and translates order-level duration predictions into project-level timing;
- an LLM-based agent layer that orchestrates these predictive services and communicates their outputs through a unified interface.

The scope is intentionally limited to decision support rather than autonomous decision-making. The system is designed to inform planners and buyers, not to replace them. Likewise, the thesis does not attempt to build a full enterprise optimisation platform; instead, it focuses on integrating a small set of high-value predictive components into an interpretable support architecture aligned with real ETO decisions.

3.4 Design Requirements

The decision context set out above, together with the research questions established in Section 1.3, gives rise to three design requirements for the proposed system.

The first concerns target formulation: each predictive component must be defined around an operationally meaningful target rather than a purely statistical one. This requirement shapes how the commodity and duration tasks are framed in Section 3.5.

The second concerns output form: the components must expose not only point estimates but also a representation of their uncertainty, so that planners and buyers can judge how far to trust each figure before acting on it.

The third concerns agent grounding: the LLM-based layer must communicate model outputs without introducing claims that those outputs do not support. This requirement governs both the architecture and the evaluation of the agent in later chapters.

Together, these three requirements connect the decision problem to the concrete design choices developed in the remainder of the thesis.

3.5 Problem Decomposition

The overall problem is decomposed into three strands. Two are predictive and cover the internal and external sources of uncertainty: manufacturing duration estimation, together with its aggregation into project-level planning, and commodity disruption forecasting. The third is architectural, concerning how their outputs are coordinated and communicated through an agent layer.

3.5.1 Commodity Risk Forecasting

The first subproblem concerns the external risk environment for nickel and copper procurement. The key design choice is not to frame this task as short-horizon price-level forecasting. As the literature suggests, such forecasts can appear accurate while offering little operational value when compared with naive baselines on noisy series (Beck et al., 2025). For procurement support, the more relevant question is whether a materially adverse or favourable move is likely to occur within a decision horizon that matters operationally.

For that reason, the commodity task is formulated in this thesis as an eight-week disruption forecasting problem. For each metal, the system estimates both the probability of a material directional move within the next eight weeks and the expected maximum magnitude of that move.

Eight weeks matches the company's medium-term procurement planning window; long enough to reposition a purchase, while short enough to remain a reliable signal. This formulation better matches medium-term procurement decisions than next-period price prediction and enables the design of alert policies tied to practical response windows.

3.5.2 Manufacturing Duration Prediction and Project Planning

The second subproblem concerns predicting the realised duration of manufacturing orders in working days. In the studied company, the operation-level planned routing hours provide a structured baseline that can be reconstructed from the ERP data. This mechanical baseline should not be confused with the full human planning process, but on its own ignores current congestion, queue state, component availability, and project context, which existing practice captures, if at all, only through the coarse load proxy noted above. The thesis therefore treats duration prediction as a supervised learning problem in which static order descriptors are complemented by richer dynamic shop-floor state features that go beyond a single summed open-hours figure.

These order-level estimates are not the end goal, however: orders rarely run in isolation, so they must be aggregated into project-level timing under shop-floor sequencing, resource contention, and component availability, which is the role of the project planning component.

3.5.3 Agent-Based Integration

The third subproblem is architectural. Even if the duration and commodity models are individually useful, they still create coordination overhead when planners and procurement staff must consult separate tools and manually reconnect their outputs to the same project context. The thesis therefore frames the LLM-based agent as an orchestration and explanation layer: it selects the relevant tools, coordinates their outputs, and communicates the resulting picture of uncertainty in natural language while remaining grounded in the underlying services.

3.6 Evaluation Framework

The evaluation strategy follows the decomposition above. The predictive components are evaluated quantitatively on their own terms: the manufacturing duration pipeline through held-out prediction accuracy, calibration-oriented diagnostics, and subgroup robustness; and the commodity pipeline through out-of-fold classification, alert utility, regression, and calibration metrics. The agent layer is evaluated differently. Because it does not generate primary predictions itself, its role is assessed through automated tests and qualitative analysis of whether it can correctly invoke tools, preserve grounding, and communicate uncertainty in a way that is coherent and useful for the intended user.

Taken together, this problem formulation defines the bridge between the literature reviewed in Chapter 2 and the implementation choices described in Chapter 4. The next chapter details how the predictive components and the agent architecture are built.

Chapter 4

Design and Development of the Decision Support System

This chapter describes the design and implementation of the proposed decision-support system. It first presents an overall view of the system architecture and the way its modules interact, and then details each module in turn: the commodity disruption pipeline, the manufacturing-order duration predictor, the project-planning layer built upon it, and the LLM-based agent that integrates them.

4.1 Overall System Architecture

The proposed system is organised as a set of independent predictive and planning services unified by a single conversational agent. Rather than a single monolithic model, each source of uncertainty identified in Chapter 3 is handled by a dedicated module with a well-defined input and output contract: a commodity disruption pipeline, a manufacturing-order duration predictor, and a project-planning layer that builds on it. Each module is computationally self-contained and can be developed and tested independently of the others. A separate LLM-based agent sits above them as an orchestration and explanation layer: it exposes each module as a callable tool, decides which tools to invoke and in what order, and communicates their combined outputs to the decision-maker in natural language. This separation of concerns keeps every predictive component independently testable while concentrating tool selection, result synthesis, and user interaction in a single place.

Figure 4.1 summarises the resulting architecture as four layers. At the base, a data layer aggregates the heterogeneous inputs the system draws on: manufacturing-order and routing records from the case company's ERP system, London Metal Exchange (LME) price and inventory series, macroeconomic and cross-asset indicators, and news signals derived from the Global Database of Events, Language, and Tone (GDELT) Global Knowledge Graph (GKG). Above it, the module layer hosts the three analytical components. The duration predictor and the project-planning layer form a tightly coupled manufacturing-side pair: the planner consumes the predictor's order-level duration estimates, together with a shop-floor simulation, to derive project-level schedules. The commodity disruption pipeline is a standalone supply-side component and does not feed the

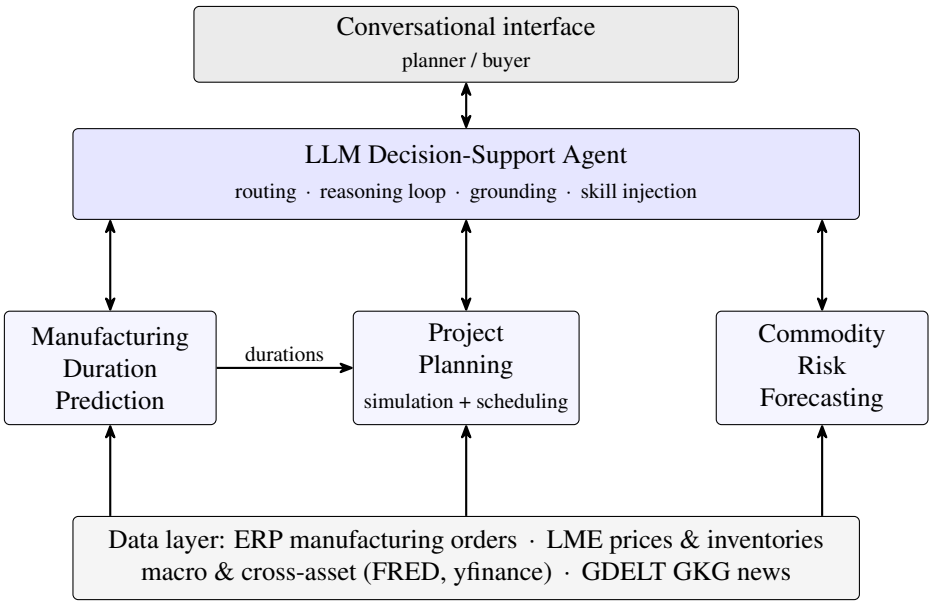

Figure 4.1: Overall architecture of the decision-support system: three predictive and planning modules on a shared data layer, exposed as tools to an LLM-based agent behind a conversational interface.

schedule directly. The agent layer sits above the modules and exposes each as a tool, deciding which to call and how to combine their outputs while remaining grounded, so that every reported figure is traceable to a specific tool call. The topmost layer is the conversational interface through which a planner or buyer queries the system.

This architecture follows directly from the problem decomposition in Chapter 3. The three predictive and planning modules together characterise the order-duration, project-planning, and commodity supply-risk uncertainties examined in RQ1, while the agent layer realises the coordination and communication concern of RQ2. The remainder of this chapter details each module in turn: Section 4.2 presents the commodity disruption pipeline, Section 4.3 the manufacturing-order duration predictor, Section 4.4 the project-planning layer built upon it, and Section 4.5 the LLM-based agent that integrates them.

4.2 Commodity Risk Forecasting Module

This section describes the supervised learning pipeline used to forecast medium-term upside and downside movements in nickel and copper prices. Weekly panels are constructed from market, macroeconomic, news, and auxiliary model signals, and a separate disruption model is trained for each metal. Copper is modelled first, and its leakage-safe disruption signal is reused as an input to the nickel model; this cross-asset link is detailed in Section 4.2.

Data Sources

All series are aligned on a weekly calendar with Friday as the anchor day. For each metal we construct a weekly panel that combines (i) cash, three-month futures prices and LME inventory indicators (London Metal Exchange, 2026), (ii) macroeconomic and cross-asset series from Federal Reserve Economic Data (FRED) and yfinance (Aroussi, 2026; Federal Reserve Bank of St. Louis, 2026) (foreign exchange (FX) rates, iron ore prices, oil, yield curve, credit spreads, China leading indicators), (iii) news-based variables derived from the GDELT GKG, including article counts and tone for metal-related entities and locations, and (iv) probabilistic price forecasts produced by Chronos, a pretrained probabilistic time-series model that generates distributional forecasts by treating discretised price sequences as token sequences (Ansari et al., 2024).

Feature Engineering

From these series we engineer a feature set grouped into six families: price and volatility, term structure and inventory, macro and cross-asset, news topics, cross-metal transfer, and Chronos-based forward risk indicators. Within each family we use standard feature transformations such as multi-horizon returns, rolling volatility, z-scores, percentile ranks, regime flags, and simple interactions (for example, volatility interacted with news sentiment). The macro and cross-asset family additionally includes cross-commodity spread signals (ratios and relative momentum between the target metal and coal, gold, battery-metal proxies, and stainless-steel indices), which serve as leading indicators of demand substitution, energy-cost pressure, and risk-off sentiment.

The cross-metal transfer family for nickel also includes the out-of-fold (OOF) predicted disruption probabilities and expected maximum returns produced by the copper model, together with derived momentum and regime signals (week-over-week probability changes, four-week lags, a directional divergence measure, and a binary stress-regime flag) and the Chronos-based probabilistic forecasts for copper, all constructed in a leakage-safe manner as described in Section 4.2.

The Chronos-based forward risk indicators family derives, for each week, an eight-week forward path distribution from a rolling context window of historical cash prices, from which we extract the P10, P50, and P90 horizon returns, the estimated disruption probability, the inter-quantile uncertainty width, and the path skew as model-agnostic complements to the backward-looking price and volatility features.

News Features From the GDELT GKG

To capture information about supply disruptions, policy interventions, and market sentiment that may not be visible in price data alone, we derive news-based features from the GDELT GKG (GDELT Project, 2015). The GKG applies natural language processing to global online news, extracting for each article a set of themes, locations, organisations, and tone scores that quantify its emotional valence and polarity.

We download GKG records for the full sample period and filter them to articles whose themes and entities are related to nickel, copper, and base metals more broadly (e.g. mining, smelting,

sanctions) and to locations and organisations that are structurally important for metal supply (such as Indonesia, the Philippines, and major producers). For each day we aggregate the filtered records into counts and average tone measures; daily aggregates are then resampled to the weekly calendar used in the rest of the pipeline.

From these weekly aggregates we construct a compact set of news features:

- overall coverage intensity (log article counts) and a coverage flag;
- volume-weighted tone and polarity measures for nickel- and copper-related news;
- supply- and policy-specific tone indices, obtained by focusing on GKG themes associated with production, export policy, sanctions, and industrial accidents;
- location- and entity-specific signals (e.g. tone of Indonesia- or Russia-related coverage).

These GDELT features allow the models to react to evolving narratives about nickel and copper supply and policy, rather than relying solely on information embedded in prices and macro variables.

Validity of GDELT sentiment as a signal. The news features used in this work rely on the tone and coverage fields of the GDELT GKG. While GDELT's sentiment scores are based on classical lexicon methods rather than contemporary large language models, several evaluations suggest that they provide a reasonable approximation to more advanced or manual annotations, especially at aggregated levels.

A large-scale comparison of GDELT's tone with neurally-derived sentiment from Google's Cloud Natural Language API over 69 million news articles reports that more than 90% of articles share the same polarity (positive vs. negative), and that daily aggregated tone series reach correlations of up to $r \approx 0.82$, with most differences arising from small variations in intensity rather than direction (Leetaru, 2019). In a separate human-coding study, GDELT's binary tone labels achieved precision and recall comparable to human annotators, whose own inter-annotator agreement was around 74% (Tambe & Friedman, 2022).

Beyond these validation exercises, GDELT-based sentiment indicators have been shown to improve return and volatility forecasts in financial markets (Shen et al., 2022), and to correlate with social unrest and protest activity (Ozdemir, 2018). In this thesis we therefore use GDELT primarily as a high-coverage, real-time proxy for aggregate news tone, aggregating its scores to weekly frequency.

Market Regime Clustering via Gaussian Mixture Models

A static weekly feature vector captures the contemporaneous level of each predictor, but it does not directly encode the broader market state in which those predictors are observed. The predictive meaning of a given signal may vary materially across environments such as periods of inventory stress, elevated macro pressure, or high volatility. To provide the downstream classifier with a

compact representation of this latent market context, the pipeline augments the main feature matrix with four soft regime-membership probabilities produced by a fold-local Gaussian mixture model (GMM).

The regime module is fitted separately inside each walk-forward fold using only data available before the fold boundary, with fold-local scaling and replay artifacts used to reproduce validation-window probabilities without look-ahead leakage. The production design uses a predefined candidate set of market-state inputs, selects configurations using both downstream precision-recall area under the curve (PR-AUC) and health diagnostics that reject collapsed or near-categorical clusters, and applies posterior softening so that regimes enter the classifier as graded contextual features rather than hard labels. The fitting sample is anchored from 1 January 2019 onwards to reflect the modern commodity-market regime; the construction details, health gates, softening transform, and sensitivity checks are reported in Appendix B.2 and Appendix B.8.

Label Construction

We formulate two types of prediction targets over a fixed forecast horizon of $H = 8$ weeks:

Directional disruption labels. Binary classification labels indicate whether the metal experiences a disruptive move over the next eight weeks. For each week t, let p_t denote the weekly cash (settlement) price and define

$$\mathscr{C}_t^{(d)} = \mathbf{1}\left[\exists k \in \{1,\dots,H\} : d \cdot \left(\frac{p_{t+k}}{p_t} - 1\right) \geq \tau\right], \quad d \in \{-1,+1\}$$

The downside label is set to 1 if at any point in the next eight weeks the price drawdown reaches the set threshold, and 0 otherwise. The upside label is defined symmetrically for upward moves.

Thresholds were calibrated independently for each commodity so that the positive rate of the disruption label averages approximately 25% across directions for both nickel and copper, ensuring a consistent definition of disruption frequency across markets and sufficient minority class representation for reliable classifier training. This yields $\tau = 0.10$ for nickel and $\tau = 0.075$ for copper, reflecting the higher realised volatility of nickel relative to copper.

Regression targets. In parallel, continuous regression targets encode the maximum directional returns realised over the next eight weeks:

$$R_{t,k} = \frac{p_{t+k}}{p_t} - 1, \quad k \in \{1,\dots,H\}$$

$$R_t^{(d)} = \begin{cases} \min_{k=1}^{H} R_{t,k} & d = -1 \\ \max_{k=1}^{H} R_{t,k} & d = +1 \end{cases}$$

These targets allow the model to learn the expected maximum magnitude of downside and upside moves, rather than their occurrence relative to a fixed threshold alone. The last eight weeks of the sample have undefined labels and are excluded from training.

Model Training and Evaluation

Cross-asset feature stacking. As introduced above, the two metals are not modelled in isolation. Copper is trained first, and its OOF disruption probabilities are carried into the nickel model as cross-asset predictive features, allowing nickel to condition on an independently derived disruption signal through the shared macroeconomic drivers and cross-commodity demand linkages that connect the two markets. This is a form of cross-asset feature stacking rather than transfer learning: no model weights are shared or initialised from the copper model; only its leakage-safe predictive outputs enter the nickel feature set.

Model structure. For each metal, the classifier stage follows a two-round cross-aware training procedure before the regression stage is fitted.

Round 1: Baseline directional classifiers. Separate Extreme Gradient Boosting (XGBoost) (Chen & Guestrin, 2016) classifiers are trained independently for the downside and upside disruption labels using the base feature set. Walk-forward cross-validation produces leakage-safe OOF probabilities $\hat{p}_d^{(0)}(t)$ for each week t and direction $d \in \{\text{down}, \text{up}\}$.

Cross-aware feature construction. The Round 1 OOF probabilities are used to construct cross-aware inputs: the downside classifier's OOF probability is appended to the upside model's feature set, and vice versa. Derived signals, such as week-over-week probability changes, four-week lagged probabilities, and a directional divergence measure are materialised at this stage, since a tree model operating on a single-timestep snapshot cannot reconstruct them internally. This allows each classifier to condition on the other direction's assessed disruption risk before being finalised.

Round 2: Final twin classifiers. Both classifiers are retrained from scratch on the augmented feature set using the same walk-forward scheme, yielding final OOF probabilities $\hat{p}_d^{(1)}(t)$. It is these Round 2 OOF probabilities, not the Round 1 baseline estimates, that are carried forward.

Regressor injection. The Round 2 OOF probabilities $\hat{p}_d^{(1)}(t)$ are injected as additional features for the corresponding regression model: the downside regressor conditions on $\hat{p}_{\text{down}}^{(1)}(t)$ and the upside regressor on $\hat{p}_{\text{up}}^{(1)}(t)$. Because each OOF value was produced by a classifier trained exclusively on data prior to week t, this enrichment introduces no look-ahead leakage. The downside and upside maximum-return targets are then modelled separately with XGBoost regressors.

Architecture validation. The two-round cross-aware twin structure described above was validated against two simpler alternatives: a single-stage variant using only the Round 1 classifiers, without cross-aware features, and a single joint multiclass model predicting {none, down, up}, with the walk-forward folds and alert policy held fixed so the only attributable difference was

model architecture. The two-round design achieves the highest downstream alert utility, justifying its additional training complexity; the full comparison is reported in Appendix B.10.

Feature selection and weighting. Feature sets are refined using SHAP-based (Lundberg & Lee, 2017) importance pruning and periodically re-estimated during walk-forward training. Regression training applies a two-component sample weighting scheme: a relevance weight that sharply upweights weeks whose realised directional return exceeds a disruption threshold ($\tau = 0.10$ for nickel, $\tau = 0.075$ for copper), and a magnitude weight that further amplifies training on larger moves. The full weighting formulation is given in Appendix B.1.

Walk-forward temporal cross-validation. To respect the time-series structure, the model is evaluated using a walk-forward scheme. In each fold, a contiguous historical window is used for training and the subsequent block for validation, yielding OOF predictions for most weeks in the historical period. Older observations remain in the training set but are down-weighted using an exponential decay on the sample weights with a learnable rate λ, reflecting the assumption that recent market regimes are more informative for near-term disruption prediction than further historical ones. For each training fold, class imbalance is addressed via XGBoost's native upweighting of the minority class proportionally to its scarcity in the training fold.

Hyperparameter selection via alert-utility optimisation. Model hyperparameters are selected by maximising a downstream reference alert utility $U^{\text{ref}}_{\text{total}}$ rather than a classification surrogate such as PR-AUC. This utility rewards correctly timed directional capture while charging activity, re-firing, adverse-week, and instability costs, and it is the objective used to select the reference alert policy reported in Chapter 5. A stricter ex-post variant that adds a full adverse-variation charge for non-materialising alerts is reported separately as a robustness diagnostic, without retraining the policy.

The utility function is structured in two layers: a *structural layer* encoding fixed domain priors about alert quality (minimum capture gate, re-fire penalty, and a penalty ceiling), and a *cost layer* that quantifies how mistimed or excessive alerts erode their practical value. The full specification of the reference utility and the stricter post-hoc re-score is given in Appendix B.1.

Hyperparameter search is carried out using Bayesian optimisation (Optuna) (Akiba et al., 2019), which co-optimises XGBoost tree hyperparameters, alert policy parameters (probability threshold, temporal decay rate ρ, minimum probability delta, and high-probability override; defined in Appendix B.1), and cost parameters jointly against $U^{\text{ref}}_{\text{total}}$ over an internal held-out window. Parameters are periodically re-tuned within the walk-forward scheme as the training window grows. A separate Optuna routine selects regression hyperparameters by minimising MAE.

Integration into the agent. The final trained models are wrapped as callable services within the AI agent. Given a current week the agent retrieves the relevant model, queries the downside and

upside classifiers and regressors, and exposes the resulting disruption probabilities and expected move magnitudes to the user in natural language.

4.3 Manufacturing Duration Prediction Module

This section describes the methodology used for manufacturing-order duration prediction. It covers the construction of the ERP-based modelling dataset, the target definition, the feature set, and the training and evaluation procedure used for the duration models.

Data Sources

The target variable and all structural features for manufacturing-order duration prediction are sourced from the case company's ERP system. For each completed manufacturing order, the following information is extracted: the order identifier and project type, the produced part code, the launch date, the first planned operation start date, the sequence of workstation codes per operation, the number of unique workstations and operations, the number and family membership of allocated components, the planned quantity per component family, and the per-operation planned hours.

The dataset is partitioned into a training set (80%) and a held-out test set (20%) by random sampling with a fixed seed. This split is used to estimate same-population serve-time accuracy rather than prospective temporal drift. To avoid look-ahead leakage, all history-dependent features are constructed causally at the predicted order's start date, with chronological shifts applied where required; each record's feature vector therefore contains only information that would have been available at serve time. Because a random split can mix observations from the same calendar period across training and test sets, temporal robustness is assessed and validated separately through the chronological test-set split reported in Appendix C.7. The target variable is the effective duration in working days, computed as the number of business days between the recorded start and end dates.

Target Variable Preprocessing

Because production start and end dates are manually entered by operators into the ERP at the time of order registration and closure, the target variable is subject to several systematic recording errors. A dedicated data-quality step excludes records exhibiting any of six categories of recording violation such as: operation sequence inversions, administrative closing delays, ghost production after interruption, parent-child date inconsistencies, and isolated close-out operations. This filtering therefore removes 3,478 of the 20,152 completed orders launched in the study period.

A further scope restriction excludes orders whose effective duration exceeds 22 working days, approximately one calendar month. This is a mostly operational, rather than a statistical, boundary: the duration estimator is a short- to medium-horizon shop-floor predictor whose feature set

captures congestion, queue state, routing, and workstation throughput, whereas orders extending beyond roughly one month are project-scale, governed predominantly by procurement lead times and cross-order dependencies that the ERP feature set does not observe and that are instead handled by the project-planning layer described in Section 4.4. All reported duration metrics therefore describe the under-22-working-day population, which comprises 16,245 orders. The full rule definitions and per-rule exclusion counts are given in Appendix A.1, the duration distribution of the excluded tail in Appendix A.2, and summary statistics of the modelling population in Appendix A.3.

Feature Engineering

Features are constructed from the raw ERP fields and organised into six conceptual groups. Approximately 150 features are produced in total prior to the SHAP-based pruning step described at the end of this section; the subset retained for model training is determined by that step and is smaller in practice.

Scope of work. The most direct predictors of duration are the features that describe how much work the order entails. Total planned hours, provided by the routing system as the sum of per-operation planned estimates, is included both in its raw form and log-transformed, since the relationship between planned hours and realised duration is approximately multiplicative rather than linear. Supporting features capture the number of operations, the maximum and standard deviation of per-operation hours, the ratio of the heaviest operation to the workstation's historical norm, and a flag for orders with negligible planned work. A set of token-based complexity scores characterises the diversity and weight of the workstation sequence, operation sequence, and component family list, complemented by a family-level diversity index.

Workstation characterisation. Each order's routing is mapped to a taxonomy of eight workstation types: sawing, laser cutting, bending, machining, assembly, automation, welding, and project mounting. These yield binary presence flags per type, the number of distinct types in the routing, and the share of planned hours concentrated on the primary type. Expanding historical statistics are computed over all prior orders that share the same workstation-type composition, providing the model with estimates of how long structurally similar orders have historically taken. An adjusted workstation estimate translates planned hours into an expected duration by weighting each operation's planned hours by that workstation's median plan-to-actual throughput ratio, computed over a rolling window of recent completed operations. To keep this ratio representative of current factory conditions, the preferred window is three months; when a workstation has fewer than 30 completed operations in that period, the window expands adaptively to six or twelve. Derived statistics capture the spread and tail behaviour of this estimate across operations.

Shop-floor load at launch. Features in this group capture the congestion state of the factory at the moment the order is launched. The concurrent order count, total concurrent planned hours

by workstation, and the number of other open orders sharing at least one workstation all reflect how much competing demand the order will face. A bottleneck load ratio normalises the load on the most constrained workstation relative to its historical median throughput. Trailing load features aggregate workstation-level planned hours over rolling ten- and twenty-two-working-day windows prior to the order start date. A queue pressure ratio divides the total concurrent load by the adjusted workstation estimate, providing a size-independent congestion signal.

Historical duration signals. Three sources of look-ahead-free historical evidence are incorporated. First, the ERP-generated queue-position field associated with the first operation at launch is mapped to an ordinal bracket and enriched with historical duration statistics computed from prior orders in the same bracket. Although this field appears to reflect queue state, it is treated here as an informative system-generated signal rather than as a directly interpretable measure of the physical workshop queue. Second, the working-day gap between the order launch date and its first planned operation start is similarly bracketed and enriched with bracket-level duration statistics, capturing how urgency at launch has historically related to actual outcome. Third, for orders producing a previously seen part, expanding statistics over all prior orders producing the same part are included: a Bayesian-smoothed mean, P75, P90, standard deviation, and observation count. All three sources are computed with a one-step chronological shift to prevent self-leakage from the current observation.

Order context and position. The project type, distinguishing stock replenishment, kanban-driven, direct sales, internal, and external client orders, is encoded ordinally to reflect the differing levels of queue tolerance and urgency associated with each category. Hierarchy features describe the order's position in the manufacturing order tree: its depth, whether it is a root or leaf node, the number of children and siblings, and aggregate planned hours and complexity figures for the parent and child subtrees. A binary flag indicates whether the routing contains any outsourced operations.

Component availability. Two binary heuristic flags are constructed from ERP status fields to indicate potential material-availability risk, namely whether any allocated component is recorded as absent from the warehouse or lacks a confirmed internal transfer at the time of feature computation. Complementing these signals, historical duration statistics derived from prior orders containing the same component references and component families are used as proxies for how component-related complexity may have influenced duration in past cases. These proxies are limited by the granularity of the ERP coding scheme: components that are operationally similar may still be represented by distinct references, so the historical features primarily capture exact code recurrence rather than broader component similarity.

SHAP pruning. Not all engineered features are retained for training. Following an initial model fit, features are pruned using SHAP-based importance scores: any feature whose mean absolute

SHAP contribution falls below one percent of the maximum across all features is discarded. This threshold removes features that contribute negligible explanatory power while avoiding the arbitrary selection decisions of filter-based methods. The retained feature set is used for all subsequent training and evaluation steps.

Model Training and Evaluation

The duration prediction model follows a two-stage architecture. The first stage is a binary XGBoost classifier that determines whether a given order has a non-trivial duration, with trivial orders defined as those starting and finishing within the same day. The second stage is an XGBoost regressor that estimates the actual duration for orders the classifier routes as non-trivial. Both stages are trained using five-fold cross-validation with randomly assigned folds.

Four hyperparameters for all three models, maximum depth, learning rate, minimum child weight, and column subsample, are selected by grid search under this cross-validation, optimising the per-model criteria described below; the search grid and selected values are reported in Appendix C.3. The L1 and L2 regularisation strengths are held at fixed per-model values rather than re-searched. These hyperparameters are then held fixed for all subsequent training and for the feature-group ablation of Appendix C.4.

For the classifier, hyperparameter selection is guided by the F-beta score on the non-trivial class with beta set to 1.5, weighting recall more heavily than precision. This asymmetry reflects the operational cost structure: misclassifying a non-trivial order as same-day is more consequential than the reverse, since it bypasses the regressor entirely and returns a same-day estimate regardless of the order's actual complexity. Within each cross-validation fold, a decision threshold sweep is performed from 0.10 to 0.90 in steps of 0.01, selecting the threshold that maximises the fold's F-beta score. The mean optimal threshold across folds is stored and applied at inference time.

The regressor is trained on the log-transformed duration target. Working in log space makes the regression loss approximately symmetric in proportional terms, preventing the model from being dominated by the few very long orders in the training data. The selection criterion is day-space mean absolute error, computed by applying the lognormal bias correction and inverse transformation to the cross-validation predictions before evaluating the loss. This correction, which adds half the in-fold variance of the log-space residuals before back-transformation, accounts for the Jensen's inequality gap between the expected log-duration and the log of the expected duration. The number of trees is not grid-searched but determined automatically per combination via early stopping, with a patience of 50 rounds.

Alongside the point estimate, a residual-quantile model is trained to produce P10, P50, and P90 uncertainty intervals around the regressor output. Rather than predicting duration quantiles directly from the raw feature vector, this second-stage model is trained on out-of-fold residuals, defined as realised duration minus the point prediction produced by the duration regressor. The residual model is again an XGBoost regressor with the quantile (pinball) loss objective (Koenker & Bassett, 1978), but its input space is augmented with the point prediction itself, low-duration

indicator features, and the classifier's non-trivial probability and margin relative to the routing threshold.

This allows the uncertainty layer to condition not only on order characteristics, but also on whether a case lies close to the classifier boundary between same-day and non-trivial orders. Hyperparameter selection for the residual-quantile model is guided by the mean pinball loss across all three quantile levels on the held-out validation fold.

Training a separate model on held-out residuals rather than calibrating the intervals analytically is a deliberately lightweight alternative to conformal interval methods such as conformalized quantile regression (Romano et al., 2019), which provide finite-sample coverage guarantees only under exchangeability assumptions that do not hold for this temporally evolving shop-floor data.

Serve Time

At inference time, the server exposes a prediction endpoint that accepts a manufacturing order identifier, fetches the corresponding row from the ERP, and constructs the feature vector using the same feature engineering logic as the training pipeline. The workstation throughput ratios used in feature construction are refreshed from the live database at startup rather than read from a static snapshot, ensuring that the adjusted workstation estimate reflects current factory conditions. An additional endpoint supports what-if queries, in which the caller supplies overrides to either the raw order fields or the engineered features; when raw field overrides are provided, the full feature construction routine is re-executed before scoring.

Prediction follows the two-stage architecture described in Section 4.3. The classifier first determines whether the order is non-trivial; orders classified as trivial are expected to start and complete within the same working day and are returned immediately with a same-day estimate. Orders classified as non-trivial are passed to the regressor, which predicts the log-transformed duration. The lognormal bias correction is applied before the inverse transformation to recover the point estimate in working days.

The residual-quantile model then predicts lower, median, and upper residual quantiles, which are added back to the point estimate to produce the final P10, P50, and P90 bounds. A post-hoc consistency clip enforces that the upper bound is no lower than the point estimate and the lower bound is no higher. Finally, a small confidence-aware downward adjustment is applied to very short regressor-routed cases, using the classifier margin and point-prediction scale to reduce systematic overstatement of uncertainty bands near the same-day/non-trivial boundary while preserving interval width.

For each prediction, SHAP contributions are computed and converted from log-space into working-day terms, producing a feature-level attribution decomposing the gap between the prediction and the population baseline. This attribution is surfaced by the agent in natural language, allowing a decision-maker to understand not only the predicted duration but which characteristics of the order are driving it.

4.4 Project Planning Module

Simulation Model

Overview and design rationale The simulation model is a discrete-time, working-day-step simulator of the shop floor, designed to answer a single practical question: given the current state of the factory, on which day should each candidate manufacturing order start? The model is not an optimisation solver and does not attempt to find an optimal schedule. This deliberately heuristic design prioritises computational speed and interpretability, enabling the model to be called at decision time as a live planning tool rather than as an offline batch computation.

The simulation operates at single working-day resolution and uses an adaptive horizon that extends until every candidate has been scheduled. All state variables, including workstation loads, material stock levels, and queue backlogs, are updated at each daily tick. The resulting state snapshots are recorded and passed downstream to the duration predictor and start-date scoring model.

Population entering the simulation Each run draws on two order populations. The candidate set contains the open manufacturing orders for which start-date recommendations are sought, selected by order identifier or project reference and represented through their operation routing, workstation sequence, planned hours, and current operation state. The background population contains all other open shop-floor orders. Background orders already in progress initialise the active load state, while open but not yet started orders enter the waiting queue and compete for capacity. The initial workstation load is seeded from a live query at the planning date, so the simulation starts from the current state of the shop floor rather than an empty-factory baseline.

Because the horizon can extend over several working weeks, the simulator also injects synthetic arrivals derived from recent workstation-level arrival profiles. This prevents simulated load from declining unrealistically as existing orders complete. Candidate and background orders then advance through their routings via fractional-day operation mechanics, with produced material credited to the simulated stock register when an order finishes. The operation-time representation, daily advancement rules, and synthetic-arrival construction are detailed in Appendix D.

Material and capacity eligibility Before a candidate may be scheduled on a given day, the simulator checks whether the required material quantities are available in the simulated stock register. The stock register evolves as background orders finish and as confirmed purchase-order arrivals are injected on their scheduled delivery dates, allowing material constraints to change during the simulated horizon.

Workstation capacity bands and the throughput target The simulation regulates how much new work each workstation may absorb using capacity bands derived from recent historical throughput ratios. These ratios are recomputed from live data at each planning request, so the capacity target reflects current typical productivity rather than a fixed calibration constant. Candidates are

then ranked by urgency slack and selected greedily subject to material and capacity gates; those failing either gate remain in the waiting pool and are re-evaluated on the next simulated day. The capacity-band derivation and scoring penalties are reported in Appendix D.4, and the throughput validation in Appendix D.5.

Project Planning

From simulation output to start-date recommendation The simulation produces, for each candidate, the earliest day on which it was successfully scheduled. This day defines the feasibility floor for the start-date recommendation. A start-date scoring model then evaluates a narrow three-working-day forward window and selects the day that minimises a heuristic cost balancing urgency, bottleneck congestion, concurrent queue size, and duration risk. The weights were set by iterative adjustment guided by domain knowledge and are treated as reasonable operating points rather than claimed optima; the full cost function is reported in Appendix D.4.

Duration prediction integration The recommended start date is passed to the duration predictor described in Section 4.3 as the planned start date for that order. The predictor constructs the feature vector using the factory state snapshot recorded on that day as the dynamic concurrent load features, ensuring that the predicted duration reflects the congestion conditions the order is expected to encounter at its recommended start. The predicted finish date is the primary scheduling output reported to the decision-maker alongside the recommended start date. The rounding rule used to convert predicted fractional duration into a finish date is given in Appendix D.4.

Dependency-aware lineage planning When a manufacturing order has open child orders that must be completed before the parent order can begin, the planning pipeline enforces a dependency-aware sequencing policy. The simulation is run jointly over the full child and parent set, the parent's start eligibility is deferred until all open children have finished, and the final parent start date is bounded by both the dependency floor and any procurement floor imposed by direct material availability. The parent's duration prediction is then produced using the factory state on that derived start day, ensuring that the resulting project schedule is internally consistent.

4.5 LLM-Based Decision Support Agent

The modelling components described in Sections 4.2, 4.3 and 4.4 are implemented as independent computational backends, each with its own invocation interface and structured output format. A decision-maker seeking to answer a question such as whether a given order can be started this week without disrupting ongoing work would need to query the planning backend, the duration predictor, and potentially the commodity disruption pipeline separately, and then reconcile their outputs manually. The LLM-based agent layer is introduced to remove this requirement: it provides a single conversational interface through which heterogeneous backend outputs are orchestrated,

interpreted, and communicated in natural language, without requiring the user to understand the individual model interfaces or interpret raw statistical outputs directly.

Overview and Design Principles

The agent is built around Gemma 4 31B-IT, an open-weight large language model developed by Google DeepMind (Google AI for Developers, 2026). During prototyping, the model was accessed through Google AI Studio rather than hosted locally, which avoided the computational overhead of running a model of this size on local GPU infrastructure while still allowing the development and testing of the proposed tool-using agent architecture. All subsequent references to "the agent" or "the LLM" refer to this model.

The architectural goal is to provide a single conversational interface through which a decision-maker can query the three modelling backends described in Sections 4.2, 4.3, and 4.4 without needing to understand their individual interfaces, inputs, or output formats. The agent does not perform any predictive computation itself; it decides which tools to call, in what sequence, and how to synthesise and communicate their outputs.

Four design principles govern the architecture. All quantitative claims in agent responses must originate from tool outputs rather than the model's parametric knowledge, to prevent hallucinated figures. The architecture must support multi-step queries that require coordinating calls to more than one backend. The reasoning loop must be stateless between turns except for the conversation history, so that responses remain reproducible given the same inputs. Finally, the agent must communicate predictive uncertainty and feature attribution in terms that are operationally meaningful to a decision-maker without a statistical background.

Point estimates returned by the duration predictor are accompanied by uncertainty intervals, and SHAP attributions are translated into natural language explanations of which order characteristics are driving the prediction. Example interactions with the deployed conversational interface, covering a single-order prediction, a project-planning query, and a commodity status query, are shown in Appendix E.2.

Tool Definitions

Each modelling backend is exposed to the LLM as a named function, collectively forming the tool manifest. The tools span the three predictive and planning backend domains. The commodity tools provide query interfaces over the nickel and copper disruption pipelines, supporting intents such as current alert status, status over a date range, episode summaries, and price overlay chart generation. The manufacturing order prediction tools accept an order identifier, retrieve the corresponding ERP row, build the feature vector, and return the point estimate, uncertainty interval, and SHAP attribution; a separate what-if tool supports scenario queries by accepting raw field overrides or engineered feature overrides and re-running the prediction accordingly. The planning tool runs the full shop-floor simulation and procurement resolution pipeline described in Section 4.4 and returns ordered start-date recommendations with constraint diagnostics.

The tool-use approach is preferred over fine-tuning or retrieval-augmented generation for this task because the underlying computations are deterministic and the models are already implemented as callable services. The LLM's role is orchestration, interpretation, and communication rather than computation: it translates structured model outputs into natural language explanations that are accessible to a decision-maker without statistical expertise, contextualises numerical results within the broader operational situation, and bridges heterogeneous model outputs into a coherent response. Exposing the models as tools with explicit input and output schemas enforces a clean separation between these responsibilities while ensuring that all reported values remain traceable to a specific model call.

Reasoning Loop

The agent follows a standard observe-reason-act loop. Each turn begins by constructing a messages array comprising the system prompt, the conversation history, and the new user message. The LLM is called with the tool manifest and a maximum response length. If the model returns one or more tool calls, each is dispatched to the corresponding handler, and the result is appended to the messages array before the next LLM call. This loop continues until the model produces a final text response with no further tool calls.

Tool results are serialised as compact text summaries rather than full structured payloads before being appended to the messages array. This keeps the context window manageable across multi-step queries that might otherwise accumulate large intermediate results. The full result objects are separately captured in a trace list that is passed to the front-end interface for rich rendering, such as prediction detail panels and commodity alert overlays, without affecting the LLM's context.

When a query requires coordinating multiple backends, the agent sequences tool calls naturally within a single turn.

System Prompt and Skill Injection

The system prompt defines the agent's role, the available tools and when to use each, output formatting constraints, and explicit rules against speculating on values that should come from tool calls. The formatting constraints are deliberately minimal: the agent is instructed to use lightweight structure where helpful but to avoid markdown headings, tables, and symbolic notation, in favour of plain text that renders cleanly in the conversational interface.

To keep the base system prompt concise, domain-specific operational knowledge is not embedded in it statically. Instead, a skill injection mechanism detects keywords in the user's message and, when relevant domain vocabulary is identified, prepends a structured skill document to the conversation history as an established prior exchange before the current turn is processed. This approach ensures that detailed procedural guidance is available to the LLM exactly when needed without inflating the context for unrelated queries.

Chapter 5

Evaluation and Discussion

This chapter brings together the empirical results and their interpretation for the two predictive pipelines: the commodity disruption forecasting pipeline and the manufacturing order duration prediction pipeline, together with the shop-floor simulation underlying the project-planning layer and the LLM-based agent layer that orchestrates them. The two predictive pipelines are evaluated quantitatively against held-out data and baselines; the simulator is validated separately as a state-transition model (Appendix D.5); and the agent is assessed through a narrow scripted routing and grounding check together with a qualitative discussion of its communication role.

5.1 Evaluation of Commodity Risk Forecasting

Evaluation Overview

The commodity pipeline is evaluated on OOF predictions generated by the walk-forward procedure described in Section 4.2, over the period from 01 July 2021 to 22 May 2026. The final eight weeks are excluded because the disruption labels are not yet fully observable. Evaluation covers both copper and nickel, and both downside and upside disruption tasks. Additional protocol details and metric definitions are reported in Appendix B.3.

The primary evaluation criterion is alert utility rather than classification accuracy alone, because the pipeline is intended to support procurement action through thresholded alert episodes. The headline utility reported in this chapter is the reference utility used for model and alert-policy selection (Appendix B.1), so the reported scores describe the system under the objective it was actually tuned to optimise. A stricter post-hoc re-score that charges adverse non-materialising alerts more heavily is reported separately in Appendix B.5; because that re-score is applied to the same fixed alerts without retraining, it is treated as a conservative robustness diagnostic rather than the primary reference result.

Classification, calibration, and regression metrics are therefore treated as supporting diagnostics whose role is to contextualise, rather than replace, the policy-level results, with the classification and calibration diagnostics reported in Appendix B.6. The main question is whether the pipeline improves on simple benchmarks under the reference alert-utility rule and which feature

blocks contribute to that usefulness. The learned policy is benchmarked against a no-skill prevalence baseline at the classifier level and, at the policy level, a simple momentum rule that alerts when recent price trend is strong enough; both are defined precisely in Section 5.1.

Main Results and Ablation Findings

The main empirical result is twofold. First, the disruption framing is operationally viable. The learned alert policy produces positive utility in all four tasks, meaning correctly timed directional alerts outweigh the encoded costs of maintaining and mis-timing them; but since a benchmark alert policy is itself positive in every task, positive utility alone is a weak test. The stronger evidence is that the learned pipeline improves substantially on that benchmark across all four tasks, most markedly for nickel (Section 5.1).

Second, the value of individual feature blocks is asymmetric across metals and does not track classifier discrimination in a simple way. The alert utility U_d (Equation B.1) remains the primary criterion because it is the operationally aligned proxy for procurement usefulness at deployment time, and the reference result is the policy actually selected by that criterion. The stricter fixed-policy re-score in Appendix B.5 is used to test sensitivity to more punitive alert costs.

Table 5.1: Alert utility and episode-level statistics for the OOF evaluation period. U_d is in percentage points of price return. Directional accuracy is the fraction of episodes in which the price moved in the alerted direction by at least 5% at some point during the active window. $\bar{V}_e^+$ is the mean gated correct-direction variation per episode.

Metal	**Dir.**	U_d **(pp)**	**Episodes**	**Mean dur. (wk)**	**Dir. acc.**	$\bar{V}_e^+$ **(pp)**
Copper	Down	32.5	7	7.00	0.857	11.1
Copper	Up	61.6	10	11.50	0.900	17.9
Nickel	Down	111.5	9	10.00	0.889	20.3
Nickel	Up	88.8	7	11.57	0.714	23.9

Table 5.1 reports the learned policy's reference alert utility in all four tasks. Both metals produce positive alert utility in both directions, meaning that correctly timed directional variation outweighs the activity, adverse-week, and re-fire penalties encoded in the reference alert-utility function. Positive utility is a necessary but not a sufficient indication of usefulness, since the benchmark alert policy is also positive throughout; whether the learned policy adds value over that benchmark is taken up in Section 5.1.

The absolute values should be read as protocol-level scores rather than monetary profit estimates, and the small number of alert episodes makes them indicative rather than precise estimates of long-run performance. For the intended purpose of capturing threshold-defined disruption episodes, nickel yields the stronger alert-utility profile, with total utility of 200.3 pp compared with 94.1 pp for copper. This comparison should be read in light of the metals' different volatility regimes: nickel's larger and more frequent directional moves create more disruption variation for the alert policy to capture. Appendix B.4 therefore reports leave-one-episode-out and cost-sensitivity checks for the reference alert policies.

The commodity evaluation is anchored on alert utility rather than on classifier discrimination alone because, in deployment, the model is used to trigger alert episodes whose value depends on firing frequency, duration, and the cost of false persistence, not merely to rank weeks by risk. The same logic also makes the ablation suite central to the empirical interpretation. Each variant retests the pipeline after removing one block while keeping the walk-forward protocol and alert-utility rule fixed, turning each block into a contribution test: if removing it improves utility or discrimination, then the block is being used by the fitted model but is not currently delivering stable marginal value in the final stack. A visual illustration of the alert episodes over the price series is provided in Appendix B.7.

Table 5.2 reports the main ablation results. The interpretation here is deliberately relative. The question is not whether a feature family is conceptually sensible, but whether it improves the realised out-of-fold policy once all downstream interactions are taken into account.

Table 5.2: Commodity classifier and alert-policy ablation results. Higher PR-AUC and higher U_{total} are better. ΔU is measured relative to each metal's baseline alert policy. The two regressor-only ablations that remove the disruption-focused sample weighting or the classifier-probability feature leave these quantities unchanged and are reported separately in Table B.12.

Metal	**Variant**	**Removed**	U_{total}	U_{up}	U_{down}	**PR-AUC Up**	**PR-AUC Down**
Copper	Baseline	none	94.1	61.6	32.5	0.539	0.426
Copper	Minus GDELT	GDELT	104.7	58.0	46.8	0.555	0.468
Copper	Minus Chronos	Chronos	119.7	78.8	40.9	0.512	0.349
Copper	Minus regime prob.	regime prob.	89.0	55.8	33.2	0.565	0.404
Nickel	Baseline	none	200.3	88.8	111.5	0.409	0.522
Nickel	Minus GDELT	GDELT	132.1	73.6	58.6	0.581	0.523
Nickel	Minus Chronos	Chronos	166.6	99.4	67.2	0.625	0.572
Nickel	Minus regime prob.	regime prob.	161.0	90.5	70.5	0.540	0.520
Nickel	Minus copper transfer	copper transfer	173.6	103.3	70.3	0.472	0.470

The two regressor-only variants that remove the disruption-focused sample weighting or the classifier-probability feature act only on the regression layer and are therefore reported separately in Appendix B.9. The remaining variants show two patterns consistent enough to matter. For nickel, in this evaluation window, every substantive feature family suggests it earns its place: removing GDELT ($\Delta U = -68.2$), regime probabilities (-39.3), Chronos (-33.7), or copper-transfer features (-26.7) degrades total utility relative to the baseline. This matters because the same variants do not always look worse under classifier metrics alone: Removing the Chronos family features, for example, increases PR-AUC while still reducing total alert utility.

Copper tells a different story. Its baseline policy is already useful, but removing GDELT ($+10.6$) or Chronos ($+25.6$) improves total utility, whereas removing regime probabilities (-5.1) weakens it. The implication is concrete: copper, in this evaluation window, supports the disruption framing, but its strongest deployable configuration is likely to be a pruned variant rather than the full reference stack. In this lower-volatility market, auxiliary signals appear to add unstable marginal information, whereas the latent-regime probabilities still earn their place. Taken together, the ablations show that feature retention should be governed by out-of-fold policy value,

not by conceptual plausibility or by classifier discrimination in isolation. The regime probabilities that remain useful in these ablations are also structurally well-behaved in diagnostic terms: the appendix reports temporal-coherence and transition-matrix views for both metals, showing persistent but non-degenerate latent-state dynamics rather than week-to-week label noise or near-absorbing clusters (Appendix B.7).

Model Versus Momentum Baseline

Table 5.3 compares the learned classifiers and alert policies against two simple benchmarks. At the classifier level, the no-skill baseline is the positive-class prevalence of each task. At the policy level, the benchmark is a momentum policy that fires when the trailing eight-week log-return of price exceeds a threshold, with that threshold calibrated on training history so that the benchmark's alert rate matches each task's positive-class prevalence, and with the same episode construction applied as for the learned policy. This momentum benchmark is itself positive in all four tasks.

Table 5.3: Commodity model-versus-baseline comparison. PR-AUC baseline is the no-skill prevalence baseline for each task. Alert-utility baseline is the frequency-matched momentum policy described above. Higher values are better for both metrics.

Metal	**Dir.**	U_d **model**	U_d **baseline**	**PR-AUC model**	**PR-AUC baseline**
Copper	Down	32.5	22.6	0.426	0.190
Copper	Up	61.6	10.9	0.539	0.298
Nickel	Down	111.5	28.9	0.522	0.206
Nickel	Up	88.8	22.0	0.409	0.254

The comparison is useful because it shows improvement at both levels of the pipeline. The classifiers outperform prevalence-based no-skill baselines in all four tasks, indicating that they capture genuine ranking structure rather than merely reflecting class imbalance. The same is true at the policy level: the benchmark alert policy is already positive in every task, but the learned alerting pipeline improves on it substantially, especially for nickel.

Regression-Layer Performance

The regression layer remains secondary to the alerting layer, but it is still relevant because it provides magnitude context conditional on an alert already being active. Its role is not to determine whether the pipeline is operationally worthwhile in the first place, but to indicate how large a price move the system expects once directional risk has already been flagged.

On weeks where the regressor actually fires, both nickel directions achieve positive R^2 (0.554 down, 0.463 up), and copper-up is marginally positive (0.045), meaning, in this case, the regressor explains some genuine variation in move size. Once the task is widened to the full true-disruption population, however, R^2 turns negative throughout, confirming that the model fits the magnitude of moves it is gated onto far better than it fits disruptions in general. The confidence intervals tell a similar story: nickel downside and copper upside are the strongest cases, whereas copper downside remains the clearest warning sign, combining the weakest fit with the lowest coverage.

Table 5.4: Commodity regression performance on the two evaluation populations. *Alert-firing* comprises all OOF weeks where the classifier fires an alert, including re-fires; n_{ep} is the number of distinct episodes represented. *True disruption* comprises all OOF weeks where a disruption did take place, independent of the alert policy. Positive Bias indicates systematic under-prediction of move magnitude. Coverage is the fraction of weeks where $r_{\text{true}} \in [\text{ci\_lower}, \text{ci\_upper}]$; the target is 0.80. Width is the mean interval width in percentage return units.

		Alert-firing						**True disruption**			
Metal	**Dir.**	**MAE**	R^2	**Bias**	**Cov.**	**Width**	n_{ep}	**MAE**	R^2	**Bias**	**Cov.**
Copper	Down	0.069	−0.323	−0.060	0.667	0.141	7	0.068	−1.685	−0.068	0.489
Copper	Up	0.048	0.045	−0.004	0.862	0.165	10	0.034	−0.651	0.011	0.905
Nickel	Down	0.042	0.554	−0.009	0.824	0.187	9	0.056	−1.273	−0.049	0.882
Nickel	Up	0.071	0.463	0.023	0.667	0.249	7	0.071	−0.135	0.061	0.746

Interpretation and Implications

The supporting diagnostics are broadly consistent with this interpretation. In both metals, the classifiers outperform prevalence-based baselines, and the full alerting pipeline also improves on a positive-utility benchmark policy, with nickel being more convincingly calibrated than copper. These results support the main conclusion, but they do not replace it: the practical question is not whether the probabilities are attractive in isolation, but whether they support operationally aligned procurement actions. Detailed classification, calibration, SHAP, and regressor-ablation diagnostics are reported in Appendix B.4–B.11.

Taken together, the commodity results support a positive but differentiated conclusion. Commodity disruption forecasting contributes positive utility under the proposed alert-policy evaluation rule in both metals, but the best feature configuration differs across them. Nickel benefits from the full stack, including external-news, regime, and cross-metal signals, whereas copper benefits from targeted pruning. The recurring methodological point is that classifier discrimination is an unreliable guide to operational value, which is why the evaluation is anchored on alert utility throughout.

A further consideration is that the alert policy encodes a single, fixed notion of what makes a useful alert. Its cost layer fixes one trade-off between false alarms and missed disruptions, yet in practice this trade-off is buyer-specific: a buyer who cannot afford to miss an adverse move favours a more sensitive policy, while one more concerned with acting on false signals prefers a conservative one. The parameters optimised here therefore represent one reasonable operating point rather than a universally optimal alert definition, and the same framework could be re-tuned to a given buyer's risk appetite by adjusting the cost weights.

5.2 Evaluation of Manufacturing Duration Prediction

Evaluation Overview

All results in this section are computed on a held-out test set of 3,249 completed manufacturing orders. Because the duration model is scoped as a short- to medium-horizon shop-floor predictor, these results apply to the filtered under-22-working-day modelling population described in Section 4.3. Throughout this section, the planned-hours-derived estimate refers only to a mechanical baseline constructed from the operation-level planned hours available in the ERP. It should not be interpreted as the company's full planning process or as the planner-assigned schedule, which also depends on human judgement, priorities, and manually maintained dates. Those planner decisions are not used as the quantitative baseline here because the available historical data does not support a sufficiently consistent reconstruction of them for the completed-order test set.

Three predictors are compared on the same orders. The first is the full two-stage pipeline: a classifier routes each order either to an immediate same-day estimate or to the duration regressor. The second is the planned-hours-derived estimate obtained by summing planned operation hours and converting them into working days. The third is an oracle that applies the regressor under ideal routing, and therefore provides a ceiling on the current architecture. Two further learned baselines are reported alongside these to isolate the effect of the modelling choices themselves: a regularised linear model (Ridge) on the same feature set, which tests whether the non-linear model class is warranted, and a single regressor trained on all orders without the routing classifier, which tests whether the two-stage architecture is justified.

MAE in working days is the primary criterion because it measures the typical planning error directly in the units used by schedulers. R^2, Bias, within-band accuracy, Within1d accuracy, and rMAE% are reported as supporting diagnostics. Detailed classifier diagnostics, subgroup breakdowns, temporal stability, and residual plots are reported in Appendix C.

Reference Model Versus Variants

Table 5.5 compares the full prediction pipeline against the planned-hours-derived estimate, two learned reference baselines, and the oracle on the complete test set.

The pipeline achieves an overall MAE of 1.148 working days, a 64.3% reduction relative to the planned-hours-derived estimate's MAE of 3.215 days. The planned-hours-derived estimate's negative R^2 (-0.442) confirms that this operational proxy explains less variance in actual durations than a constant mean prediction would, driven by its systematic positive bias of 3.016 days and its insensitivity to shop-floor state. The pipeline attains $R^2 = 0.712$ and a bias of 0.401 days, with within-one-day accuracy of 70.0% against 52.4% for the planned-hours-derived estimate.

Two learned reference baselines clarify which design choices drive this result. A regularised linear model (Ridge) trained on the identical feature set attains an MAE of 2.366 days ($R^2 = 0.349$); the gradient-boosted regressor more than halves this error, indicating that the mapping

Table 5.5: Overall performance on the test set ($n = 3{,}249$ manufacturing orders). InBand is the fraction of predictions within $\pm 15\%$ of actual duration. W1d is the fraction of predictions with absolute error ≤ 1 working day. rMAE% is relative MAE normalised by the training-group median duration. Bias is the mean signed error (actual $-$ predicted); positive values indicate systematic under-prediction. Ridge is a regularised linear regression and Single regressor a single XGBoost regressor trained on all orders without classifier routing; both share the pipeline's feature set and isolate, respectively, the model-class and two-stage-architecture decisions.

Model	n	MAE (d)	RMSE (d)	R^2	Bias (d)	InBand (%)	W1d (%)	rMAE (%)
PH estimate	3,249	3.215	5.582	-0.442	3.016	4.5	52.4	120.6
Ridge (linear)	3,249	2.366	3.751	0.349	-0.008	9.0	44.0	96.8
Single regressor	3,249	1.171	2.490	0.713	0.342	25.1	70.5	49.4
Pipeline	3,249	1.148	2.494	0.712	0.401	56.3	70.0	47.5
Oracle	3,249	1.054	2.441	0.724	0.485	63.6	74.0	43.0

from the engineered features to realised duration is materially non-linear and that the more expressive model class is warranted.

A single XGBoost regressor trained on the whole order population without the first-stage classifier reaches an MAE of 1.171 days, marginally above the full pipeline's 1.148 days, and is slightly higher on within-one-day accuracy (70.5% against 70.0%).

The two-stage architecture therefore yields only a small improvement in mean error over the single regressor; its primary justification is operational. Roughly 40% of orders (1,317 of 3,249 test orders) start and finish within a single working day, forming a discrete same-day mass that a continuous regressor cannot represent exactly, since it always predicts a small positive duration for such cases. The classifier identifies this mass explicitly and routes it to an exact same-day estimate, which is the answer a planner actually requires when asking whether an order will be completed on the same day it starts. This distinction is captured by within-band accuracy, which rises to 56.3% for the two-stage pipeline against 25.1% for the single regressor despite their comparable MAE: the band around a same-day order has zero width, so only an exact same-day prediction can fall inside it. The architecture thus encodes a genuine operational category rather than merely improving aggregate accuracy.

The oracle MAE of 1.054 days represents the current ceiling on regressor performance under ideal classifier routing and lies 0.094 days below the pipeline MAE. This narrow gap confirms that the dominant source of residual error is the regressor's limited capacity to fit the long-duration tail, rather than classifier routing errors. The first-stage routing itself is strong, with PR-AUC of 0.977 on the multi-day class; the detailed classifier table and precision-recall curve are retained in Appendix C.1.

Performance Heterogeneity Across Duration Brackets

Within the multi-day order population ($n = 1{,}932$ orders with a multi-day actual duration), performance varies substantially with actual duration. Table 5.6 disaggregates the key metrics by

duration bracket.

Table 5.6: Regressor performance on multi-day orders segmented by actual duration bracket ($n =$ 1,932 orders with a multi-day actual duration).

Bracket	n	MAE (d)	rMAE (%)	R^2	Bias (d)	InBand (%)	W1d (%)
1–2d (very short)	725	0.834	39.3	−7.101	−0.381	31.6	69.2
3–7d (short)	721	1.339	57.1	−1.009	0.273	44.5	54.2
8–14d (1–2w)	326	2.722	101.7	−3.260	2.161	45.4	39.0
15–22d (2–4w)	160	6.143	236.6	−12.730	6.033	31.9	19.4
Overall	1,932	1.781	72.8	0.588	0.823	38.8	54.4

For the shortest multi-day orders (1–2 days) the model is most precise, with rMAE% of 39.3% and within-1-day accuracy of 69.2%. Precision then degrades steadily with duration: rMAE% rises to 57.1% for 3–7 day orders and 101.7% for the 8–14 day range, while within-1-day accuracy falls to 54.2% and 39.0% respectively, consistent with greater inherent variability at longer durations.

For orders exceeding 14 working days ($n = 160$, approximately 8% of multi-day orders), MAE reaches 6.143 days and the bias of 6.033 days reveals systematic under-prediction. This primarily reflects uncertainty not captured by the available ERP features. Long-duration orders are more exposed to contingencies that are only partially observed in the ERP-derived features, so the current data provide a weaker basis for explaining their realised completion times. The negative R^2 values throughout the table are a mechanical artefact of within-stratum evaluation: within each narrow bracket, cross-order variance is insufficient to produce positive explained variance from a model trained on the full range, and should not be interpreted as an indicator of poor overall predictive validity.

The model's improvement is not uniform across the order population. It is strongest for the short and short-to-medium jobs that dominate the factory mix, and it weakens progressively in the long-duration tail. For orders above 14 working days, the positive bias of 6.033 days reveals systematic under-prediction, which is consistent with greater exposure to interruptions, resequencing, and project-level dependencies that are only partially visible in the available ERP signals.

The broader subgroup diagnostics point in the same direction. Workstation-type and project-type performance is heterogeneous, but the strongest errors cluster in contexts where queueing and priority shifts variability matter most. Those detailed breakdowns and residual plots, together with the temporal-stability table, are reported in Appendix C.2 and Appendix C.7. The temporal split shows no evidence of systematic degradation over the ten-month test window.

Beyond point accuracy, the duration pipeline was also evaluated through the empirical coverage of its prediction intervals, that is, how often the realised duration falls inside the predicted P10–P90 band. Intervals exist only for orders the classifier routes to the regressor, so coverage is measured on that subset. A small number of these orders were routed to the regressor but in

fact completed the same day (zero working days). They are excluded from the coverage analysis, because the regressor's interval for a multi-day job always lies above zero, so a zero-day outcome registers as a miss however well the interval is calibrated. Such a miss reflects a routing error by the classifier rather than a fault in the interval model, so these orders are instead counted where they are informative: in the classifier's own evaluation, where they are retained.

On this filtered evaluation population, the residual-quantile interval model achieves 63.4% empirical P10–P90 coverage over 1,865 orders, with mean interval width 3.21 working days.

This falls materially short of the nominal 80% coverage, and the pattern is present in every bracket and worst in the tail: coverage is fairly uniform across the sub-22-day brackets (65.7%, 64.0%, and 67.6% for the 1–2, 3–7, and 8–14 day ranges), while the 15–22 day tail remains the weakest segment at 43.0%, indicating that long and operationally volatile jobs are still underestimated even after the interval adjustment. A more detailed breakdown of interval coverage, tail misses, and interval width, together with a comparison against alternative interval-construction methods, is provided in Appendix C.5 and Appendix C.6.

Interpretation and Planning Implications

SHAP values are computed for each test observation and aggregated by feature group. Figure 5.1 reports the mean absolute SHAP contribution, expressed as a percentage of the total, for each feature group.

The largest contributor by a wide margin is the workstation characterisation group (40.4%), which bundles the expanding-window estimates of how long structurally similar orders have historically taken, the adjusted throughput ratios, and the workstation-type composition of the routing. Its dominance supports the relevance of the throughput-ratio computation described in Section 4.4: the ratio of planned hours to historical median throughput is a far stronger duration proxy than planned hours alone.

The historical duration signals group is second (21.3%), capturing the ERP queue-position field, launch urgency, and part-level history. Shop-floor load and queue state at launch contributes 14.0%, supporting the inclusion of dynamic congestion features alongside static order characteristics (Section 4.4), and component availability a further 13.1%. Congestion at launch (14.0%) still outranks raw work content: the scope-of-work group, which contains total planned hours and the complexity scores, accounts for only 5.6% of attribution, on par with order context and position (5.6%). This is not because planned hours are uninformative, but because their signal is largely absorbed by the correlated workstation-characterisation and throughput features; the planned-hours baseline, by contrast, omits exactly the shop-floor-state variables the fitted model relies on.

These shares describe how the fitted model distributes attribution, which is not the same as how much each group contributes to held-out accuracy. A leave-one-feature-group-out ablation, which retrains the full pipeline with each group removed in turn and re-evaluates it on the same held-out test set, measures this *marginal* contribution directly and is reported in Appendix C.4. It makes the distinction concrete: workstation characterisation attracts the largest SHAP share yet is close to redundant once the model is retrained without it, since its signal is recoverable from

the other groups, whereas the historical duration signals group is the most load-bearing under ablation. Attribution and marginal predictive value should therefore be read together rather than interchangeably.

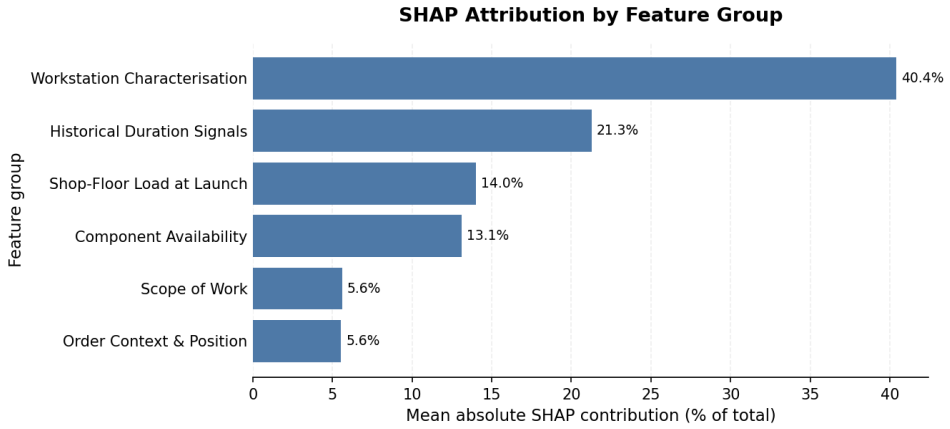

Figure 5.1: Mean absolute SHAP contribution by feature group, expressed as a percentage of the total, for the regressor evaluated on the test set. Groups are sorted in descending order of contribution.

Taken together, the manufacturing-order results show that the model's practical value comes from using the information set the planned-hours-derived estimate omits. The largest gains are not achieved by refining work-content arithmetic alone, but by representing the congestion environment into which each order is launched. At the same time, the long-duration tail remains the clearest limitation of the current approach, which is why the appendix retains the more granular diagnostic breakdowns.

5.3 Evaluation of the LLM-Based Agent

The agent layer is evaluated as a proof of concept rather than as a validated decision aid. Because it produces no primary predictions of its own, and because the intended user population at the case company is small, its assessment deliberately combines a reproducible but narrow contract check of its orchestration behaviour with a formative, qualitative evaluation by intended users. Together these establish that the architecture works as specified and give early signal on how such users react to it. Neither is designed to show that the agent improves decision quality or earns durable trust; that would require the controlled, longitudinal user study set out as future work in Section 6.4. This scope is stated at the outset so that the results below are read as formative evidence rather than summative validation.

Routing and Grounding Validation

The agent layer does not generate primary predictions of its own; its role is to route each query to the correct backend, pass valid arguments, and communicate grounded results. It is therefore as-

sessed differently from the two predictive pipelines. To make the architectural claim reproducible, the agent layer was evaluated with a scripted routing and grounding suite. The suite is deliberately narrow: it tests whether representative prompts trigger the expected backend tools, pass required arguments, avoid tools in negative-control turns, and keep quantitative statements grounded in the prompt or tool outputs. All fifteen scenarios passed these checks, including the project-planning scenarios fixed to a pre-defined project at the latest database date shop-floor snapshot. This supports the claim that the implemented agent satisfies the basic orchestration contract required by the architecture. It does not demonstrate improved decision quality, planner adoption, or optimal scheduling behaviour. The scenario specification and category-level results are reported in Appendix E.

Informal User Evaluation

Beyond the scripted routing suite, the agent was assessed in a small, formative evaluation with four members of the intended end-user population, chosen to span the decision areas the system is meant to support: two production planners, one working at the macro-planning level and one at the micro-planning level of labour allocation; a buyer also responsible for pre-procurement; and a senior manager with cross-functional oversight of the business and prior experience as a buyer.

The sessions were exploratory rather than scripted: each participant interacted freely with the agent, focusing on the capabilities relevant to their role. The two planners exercised the order-duration and project-planning tools, the buyer the commodity-risk tools, and the senior manager both. After interacting, each participant rated the agent on a five-point scale across four dimensions defined for this evaluation and made known to participants in advance: interpretability, trustworthiness, relevance, and actionability (Table 5.7). In addition to the numeric ratings, the facilitator recorded informal observations on participants' reactions and stated concerns; these are reported below as qualitative themes rather than as systematically coded findings. Given the panel size, the results are indicative and motivate directions for future work; they do not constitute a statistically powered user study.

Table 5.7: Informal expert evaluation scores (1–5) by dimension. Each column is one evaluator; the final column reports the per-dimension mean.

Dimension	E1	E2	E3	E4	Mean
Interpretability	3	4	2	4	3.25
Trustworthiness	3	3	1	3	2.50
Relevance	4	3	1	3	2.75
Actionability	4	2	3	4	3.25
Mean	3.50	3.00	1.75	3.50	2.94

The pattern of scores is consistent with the agent's intended role and with the ETO setting it is deployed into. The strongest dimensions were interpretability and actionability (mean 3.25). Together these indicate that the communication layer is doing its job: the underlying model outputs

are surfaced legibly, the agent adds explanatory value on top of them rather than merely relaying numbers, and the totality of information is presented in a form that participants felt mostly able to act on without first consulting another system. This is consistent with the communication-oriented design discussed in Section 5.5, such as the decision to translate SHAP attribution into natural language.

Trustworthiness was the lowest-scoring dimension (mean 2.50), which is the expected result rather than a surprising one. Managers and buyers typically require extended exposure to a new tool before they rely on its outputs, and this is especially true when the tool departs from the established methods and heuristics they currently use; a single moderated session is far short of the familiarisation such trust normally demands. The low score is therefore better read as a baseline adoption signal than as evidence of a defect in the predictions themselves, and it reinforces the framing of the system as decision support whose value is realised over repeated use.

Relevance (mean 2.75) reflects a mismatch of expectations rather than a failure of execution. Several participants wanted the agent to do things it was neither designed nor programmed to do, in some cases expecting it to address essentially any question a human planner might field. The implemented agent is, by construction, an explanatory and orchestration layer over a fixed set of ML models and tools, and its scope is bounded by what those backends can answer. The lower relevance score thus locates a real direction for future work: broadening the set of underlying capabilities and setting clearer expectations about scope, rather than indicating that the existing capabilities were poorly delivered.

This pattern was visible across roles: the planner working at the macro level, whose questions about timing and project completion fall within the system's backends, found the agent directly applicable, whereas the planner concerned with labour allocation, which none of the underlying tools addresses, found it least useful. Relevance therefore tracked how far each role's questions fell inside the system's bounded scope rather than the quality of delivery within it.

5.4 Discussion of RQ1: Uncertainty in ETO Decision Processes

The first research question asks how uncertainty in order durations, project planning, and commodity supply risks manifests in ETO decision processes, and what limitations current predictive approaches exhibit in practice.

The manufacturing duration results provide a direct empirical answer for the internal planning side of the problem. The planned-hours-derived estimate, which represents the structured duration signal available in the ERP rather than the full human planning process, achieves an MAE of 3.22 working days and a negative coefficient of determination ($R^2 = -0.442$), indicating that it explains less variance in realised durations than a constant mean prediction would. Its systematic positive bias of 3.02 days confirms a recurring pattern in ETO settings: work-content estimates recorded at order creation are not equivalent to realised duration predictions, because they ignore congestion, sequencing effects, and the operational state into which the order is launched.

The SHAP attribution analysis sharpens this diagnosis. The dominant predictor group is workstation characterisation (40.4% of total attribution), which encodes how long structurally similar routings have historically taken; features capturing shop-floor load and queue state at launch contribute a further 14.0%, while planned work content itself accounts for only 5.6%. This implies that, for duration in the studied environment, the model attributes far more predictive weight to historical throughput behaviour and the congestion environment an order encounters than to the amount of work specified in its routing. A leave-one-group-out ablation (Appendix C.4) refines this further: the historically derived duration signals carry the largest marginal contribution once the model is retrained, confirming that it is information beyond the static work content that drives predictive accuracy.

For the commodity side, the relevant limitation is different. Here the problem is not that planners lack any signal, but that market signals, price dynamics, and external-risk indicators are difficult to translate into a coherent procurement action frame. The evaluation framework developed in Section 5.1 reflects this by centring alert utility, calibration, and disruption-focused forecasting rather than short-horizon price accuracy alone. Even where predictive performance is acceptable, operational usefulness depends on whether the model can express risk over a decision horizon and in a form that supports action. This is precisely the gap that the commodity pipeline is designed to address.

As for the project-planning layer, it is not evaluated as an independent optimisation policy against planner decisions. Its evidential status is compositional: the duration predictor is evaluated on held-out completed orders, the simulator is backtested against point-in-time shop-floor dynamics, and the start-date recommendation logic is a deterministic heuristic that combines these evaluated inputs with explicit stock, capacity, dependency, and urgency rules. The resulting planning output should therefore be interpreted as a decision-support heuristic grounded in validated components rather than as a proven optimal scheduling policy.

Several limitations qualify this predictive picture. The duration model's systematic underperformance on orders exceeding 14 working days reflects a lack of information in the available feature set rather than a simple modelling artefact. Long-duration orders are more exposed to unplanned interruptions, opportunistic re-sequencing, and project-level dependencies whose effects are not captured in the ERP records used for feature construction. The positive bias of 6.033 days for this segment suggests that the factors driving orders beyond the two-week threshold are systematically under-observed.

A further limitation is the dataset's span of only approximately ten months. Without even a full year of data the model cannot learn seasonal patterns, and under a random split the training and test sets both fall within that same ten-month window. The temporal stability diagnostics reported in Appendix C.7 show no evidence of systematic degradation over the window, but how the model would perform on data from a different year remains an open question.

For the commodity pipeline, the most substantive limitation is that commodity forecasting remains vulnerable to regime shifts, exogenous shocks, and changes in the information content of news and macroeconomic signals. This is particularly relevant for nickel, whose strong baseline

utility was shown to depend on every feature block in the stack; that dependence may not hold under out-of-period conditions. Even when the disruption framing is better aligned with procurement decisions than short-horizon price forecasting, the practical value of the alerts depends on the model remaining calibrated as market conditions evolve.

5.5 Discussion of RQ2: Integrating Heterogeneous Predictive Tools into an LLM-Based Agent

The second research question asks how heterogeneous predictive and planning tools can be integrated into an LLM-based agent that coordinates calls to multiple models while exposing their outputs coherently and usefully to ETO decision-makers.

Several design decisions proved consequential in practice. Developing the prototype on the open-weight Gemma 4 31B-IT rather than a larger proprietary model confirmed that the architecture's tool-use patterns, prompt design, and skill injection mechanism remain viable under the capability constraints of a model that could realistically be self-hosted within the organisation's data perimeter. The tool-as-service pattern enforces the grounding constraint central to the architecture's reliability, keeping every numerical claim traceable to a specific tool call rather than to the model's parametric knowledge, consistent with Simchi-Levi et al. (2025), who position LLMs as orchestration layers over analytical tools rather than as autonomous decision-makers.

Communication itself proved to be a first-order design concern rather than a downstream presentation detail, on the premise that a planner who cannot interrogate a prediction will not act on it regardless of its accuracy. As such, the appropriate level of detail proved strongly context-dependent: single-order queries warrant explicit point estimates, uncertainty intervals, and dominant drivers, whereas project-level scheduling risk is better served by qualitative characterisations than by aggregated figures. Encoding this as contextual skill-injection guidance proved more effective than static rules in the base system prompt. The informal user evaluation (Section 5.3) offers preliminary corroboration: interpretability and actionability were the highest-rated dimensions, indicating that the communication layer largely succeeded in surfacing model outputs legibly and in an actionable form.

Several limitations qualify the evidence available for the agent layer. The two evaluations reported in Section 5.3 address complementary but partial questions. The scripted routing and grounding suite validates only a narrow integration contract, namely that the agent calls the expected tools, passes required arguments, abstains when appropriate, and keeps numerical statements grounded across a small set of representative scenarios. It says nothing about whether the resulting responses are useful to a decision-maker. The informal evaluation begins to address that gap with end-users, but it is formative rather than confirmatory: with four evaluators and a single moderated session per participant, its scores are indicative and cannot support claims about the system's value across the broader operational population.

The evaluation findings also locate the two substantive limitations more precisely than a generic appeal to "communication quality" would. First, trustworthiness was the weakest-rated

dimension, and the single-session design cannot in principle measure the quantity that matters here: whether reliance grows as planners accumulate experience with the tool against their existing methods. Trust calibration in decision support is a longitudinal property, and establishing it would require sustained use rather than a one-off assessment. Second, the lower relevance score reflects a scope boundary that is intrinsic to the architecture rather than a tuning deficiency: the agent is an explanatory and orchestration layer over a fixed set of predictive backends, and questions outside that set cannot be answered however well the communication layer is designed. Extending the underlying capability set, and setting clearer expectations about scope, is therefore a genuine direction for future work.

5.6 Managerial Implications

The combined findings of this thesis carry several implications for organisations operating in ETO environments that are considering data-driven decision support. The most immediate practical implication concerns the quality of the duration signal available at order creation time. Planned hours reflect estimated work content rather than predicted realised duration; as a result they vary across estimators, ignore current shop-floor conditions, and carry a systematic positive bias relative to actual outcomes. The duration pipeline provides a substantially more accurate and consistent alternative that reuses the planned-hours data already entered at order creation, combined with automatically available shop-floor state, and therefore imposes no new manual data entry on the planner.

The broader implication is architectural. Even strong predictive components remain difficult to use when they are distributed across separate tools and interpreted independently by different users. The agent architecture demonstrates that LLM-based orchestration of heterogeneous predictive models is technically feasible within an industrial data perimeter using open-weight models deployed locally. Exposing predictive backends as callable services with clear schemas is enough to create a conversational layer that combines duration, planning, and commodity-risk signals into a single grounded interaction.

The informal evaluation qualifies how far that technical result carries in practice: decision-makers found the combined outputs clear and actionable, but technical feasibility is not the same as adoption. Two organisational conditions emerged as decisive. Trust in a tool that departs from established planning methods is earned through sustained use rather than conferred at first contact, so organisations should expect a familiarisation period before such a system is relied upon. Because the agent answers only what its underlying backends support, its scope must be communicated explicitly to avoid the mismatch between user expectations and delivered capability that the evaluation surfaced.

That same mismatch is also an opportunity: the breadth of questions users brought to the agent indicates a demand the architecture is well positioned to meet, since each new predictive or analytical backend exposed as a callable service extends the agent's reach without altering its

orchestration and communication layer. Expanding the capability set in this way is a direct path from the present prototype towards broader operational value.

Chapter 6

Conclusions

This thesis addressed the problem of how to support ETO decision-makers who must reason simultaneously about uncertainty in manufacturing order duration and uncertainty in commodity-related supply risk. The goal set out in Chapter 3 was not merely to improve isolated predictions, but to integrate heterogeneous predictive components into a unified, interpretable decision-support architecture aligned with operational practice.

6.1 Summary of Findings

This thesis set out to support ETO decision-makers who must reason simultaneously about internal uncertainty in manufacturing-order duration and project planning and external uncertainty in commodity supply risk. It did so by designing and implementing a modular decision-support system: a manufacturing-order duration predictor, a project-planning layer built upon it, a commodity disruption-forecasting pipeline, and an LLM-based agent that exposes these services as tools and communicates their outputs through a single grounded conversational interface.

On the manufacturing side, the duration pipeline reduced mean absolute error to 1.15 working days, a 64% improvement over the planned-hours-derived estimate available in the ERP, whose negative R^2 confirmed that it explains less variance in realised durations than a constant mean.

The attribution and ablation analyses agree that planned work content adds little, but diverge on what carries the model's performance: the workstation-characterisation and throughput features dominate the SHAP attribution yet prove largely redundant under retraining, where the marginal accuracy is instead carried by historical duration signals, component availability, and current shop-floor load. What is robust across both views is that the gains come from the contextual and shop-floor-state information the planned-hours baseline omits, rather than from work content.

Accuracy was strongest for the short and medium jobs that dominate the factory mix and degraded systematically in the long-duration tail, where unobserved interruptions and project-level dependencies leave orders beyond two weeks consistently under-predicted. The project-planning layer was not evaluated as an independent scheduling policy; its output is best read as a decision-support heuristic grounded in these validated components.

On the commodity side, reframing procurement risk as eight-week directional disruption forecasting, evaluated through alert utility rather than price accuracy, produced positive and operationally meaningful alerts for both nickel and copper, improving substantially on a frequency-matched momentum benchmark and most markedly for nickel. The value of individual feature blocks proved asymmetric across the two metals: nickel benefited from the full stack of news, regime, and cross-metal signals, whereas copper was strongest under targeted pruning, reinforcing that feature retention should be governed by out-of-fold policy value rather than classifier discrimination.

Finally, the LLM-based agent demonstrated that heterogeneous predictive backends can be integrated through tool calls into a usable, grounded interface. A scripted suite confirmed that the agent routes queries to the correct backends, passes valid arguments, and keeps numerical claims traceable to tool outputs, while an informal evaluation with four intended end-users rated its interpretability and actionability most highly and its trustworthiness and relevance lowest, the former a longitudinal property a single session cannot establish, the latter a consequence of the agent's bounded scope.

6.2 Answers to the Research Questions

The findings summarised above translate into direct answers to the two research questions posed in Chapter 3; the supporting evidence and its full interpretation are given in the discussion of Sections 5.4 and 5.5.

RQ1. Across all three decision areas, uncertainty in ETO processes takes a common form: outcome-relevant information is either unavailable at decision time or fragmented and unsystematic when it is available. Current practice does not assemble that information into one reliable estimate or turn it into an actionable one.

In manufacturing duration, the company is not without signal: it holds the planned routing hours, a coarse load proxy from the summed planned hours of open orders, and the launcher's own experience of similar work. The limitation is twofold. Part of what determines realised duration, such as unplanned interruptions and re-sequencing, is simply not observable in the data at order creation; and the signal that is observable is split across these sources and never combined or calibrated against realised outcomes, so what reaches downstream planning stays informal and carries a large, uncorrected bias.

A dedicated model addresses the second problem by combining the available signals systematically, but cannot overcome the first, which is why it remains limited on the long-duration orders whose drivers lie outside the ERP data. The same pattern holds externally, where commodity supply risk is less a price-accuracy problem than a translation problem: the market signal exists, but conventional forecasting does not express it over an actionable decision horizon, and reframing the task as disruption forecasting closes that gap only as long as calibration holds when market regimes shift.

Project planning inherits the gap from both sides at once, compounding the duration model's error against a schedule that cannot be validated, which is why it is best supported by a transparent heuristic over validated components rather than an opaque optimiser.

RQ2. The integration problem proved to be one of constraint rather than capability: heterogeneous predictive models can be coordinated coherently not by making the language model more powerful, but by reducing it to an orchestration and communication layer over tools that perform the computation. Exposing each model as a callable tool with an explicit input and output schema is what enforces grounding, keeping every reported figure traceable to a specific backend call while still supporting multi-step queries that span backends within a single turn.

Within this constrained role, coherent and interpretable communication became the first-order design concern rather than a presentation detail: context-dependent guidance injected as skills, translating uncertainty intervals and SHAP attribution into operational language, proved more effective than static prompt rules. Because the model is asked to orchestrate and explain rather than compute, the architecture remains feasible under the capability constraints of an open-weight model that can be self-hosted within the organisation's data perimeter.

What the constraint does not resolve is human rather than technical: whether the interface improves decision quality and whether planners come to trust it are longitudinal properties the present evaluation could not establish.

6.3 Contributions and Limitations

Contributions

This thesis makes three main contributions. First, it clarifies the structure of decision support in ETO environments by framing internal manufacturing uncertainty and external commodity risk as a single integration problem rather than a set of isolated prediction tasks. Second, it proposes and implements an agent architecture for multi-model decision support in which an LLM coordinates specialised predictive services rather than replacing them, realised as a working prototype grounded in industrial data and respecting practical deployment constraints, including the use of an open-weight model compatible with local hosting. Third, it provides an empirical basis for the manufacturing duration pipeline and a procurement-aligned evaluation framework for the commodity pipeline anchored on alert utility, together with design implications for the use of LLM-based agents in industrial decision support.

Limitations

Several limitations qualify these contributions. On the predictive side, the duration model degrades on orders beyond roughly two weeks, whose drivers, unplanned interruptions, re-sequencing, and project-level dependencies, are only partially observable in the ERP features; its prediction intervals also show undercoverage, most severely in that same tail. The dataset spans only about

ten months, so seasonal effects cannot be learned and out-of-year generalisation remains untested. The project-planning layer relies on domain-tuned heuristics rather than formally optimised policies and is not evaluated against a ground-truth schedule.

On the commodity side, the forecasts remain exposed to regime shifts and changing signal quality. This is a particular concern for nickel, whose strong performance depends on every feature block in the stack. The alert policy also encodes a single fixed notion of alert quality, whereas what counts as a useful alert depends on a buyer's risk appetite, so the parameters used here represent one operating point rather than a universal optimum.

Most significantly, the agent layer was assessed only through a scripted contract check and an informal, formative evaluation with four end-users in single sessions: this validates the orchestration contract and gives preliminary feedback, but provides no statistically grounded evidence that the interface improves decision quality or trust for the intended operational users.

A final consideration concerns how far these findings extend beyond the single industrial case in which the system was built. The three components generalise to different degrees, and it is worth separating the specific artefacts from the underlying approach.

The duration model is the most case-bound component: it is fitted to one company's ERP schema, workstation mix, and roughly ten-month order history, so its parameters and error profile would not carry over directly. Its modelling logic, however, is general: predicting realised duration from the historical durations of comparable work and the concurrent shop-floor load an order meets at launch, rather than from planned work content alone, is an approach any ETO producer with comparable ERP records could re-apply by retraining on local data.

The commodity pipeline is the most broadly transferable: it is trained on public market, news, and regime signals rather than on company data, so a firm exposed to the same metals could adopt the trained models directly. Only the eight-week horizon and the alert-policy operating point are local choices, reflecting this firm's procurement window and risk appetite; another buyer would retune them.

The agent layer is general by construction: orchestrating tool-wrapped predictive services under a grounding constraint is independent of which tools sit behind it, and the same pattern would accommodate a different set of backends in a different organisation.

What remains genuinely specific, then, is the duration model's fitted parameters and the handful of operating choices tuned to this company; the methodological framing and the integration architecture are not tied to the case and are the part of this work most likely to transfer. Confirming that transfer in another setting is left to future work.

6.4 Future Research

Several directions follow naturally from this work. For the manufacturing side, the most promising path is data enrichment aimed at better representing the factors that drive long-duration orders and project-level dependencies. A related extension is richer uncertainty propagation from individual-order predictions to project-level completion risk. For the commodity side, future work should

stress-test calibration under changing regimes, and refine the link between alert thresholds and procurement policy. A natural extension is configurable, buyer-specific alert policies that expose the false-alarm/missed-disruption trade-off as a tunable risk-appetite setting.

For the agent layer, the most important next step is a controlled user study with domain experts that builds on the informal evaluation reported here. Such a study should evaluate whether the agent improves decision quality relative to direct exposure to model outputs, track the development of trust through sustained use rather than a single session, and identify what forms of explanation, uncertainty communication, and interaction actually help planners in practice.

A complementary direction follows from the breadth of questions users brought to the agent, which exceeded its current scope: because each predictive or analytical capability is exposed as a callable service, the architecture can be extended with new backends without altering its orchestration and communication layer, offering a direct path towards broader operational value. Over time, this could also support a more adaptive response layer that learns from interaction feedback while preserving the grounding guarantees of the current tool-based design.

In summary, the thesis argues that predictive decision support in ETO manufacturing becomes substantially more useful when prediction, interpretation, and cross-model reasoning are treated as a single design problem. The implemented prototype does not solve that problem completely, but it demonstrates a viable architecture for addressing it and establishes a strong basis for future refinement.

Bibliography

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2623–2631. https://doi.org/10.1145/3292500.3330701

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Pineda Arango, S., Kapoor, S., Zschiegner, J., Maddix, D. C., Wang, H., Mahoney, M. W., Torkkola, K., Wilson, A. G., Bohlke-Schneider, M., & Wang, Y. (2024). Chronos: Learning the language of time series. *Transactions on Machine Learning Research*. https://arxiv.org/abs/2403.07815

Aroussi, R. (2026). `yfinance`: Download market data from yahoo finance [Open-source Python package; not affiliated with Yahoo]. https://pypi.org/project/yfinance/

Baryannis, G., Dani, S., & Antoniou, G. (2019). Predicting supply chain risks using machine learning: The trade-off between performance and interpretability. *Future Generation Computer Systems*, *101*, 993–1004. https://doi.org/10.1016/j.future.2019.07.059

Beck, N., Dovern, J., & Vogl, S. (2025). Mind the naive forecast! a rigorous evaluation of forecasting models for time series with low predictability. *Applied Intelligence*, *55*, 395. https://doi.org/10.1007/s10489-025-06268-w

Burggräf, P., Wagner, J., Koke, B., & Steinberg, F. (2020). Approaches for the prediction of lead times in an engineer to order environment—a systematic review. *IEEE Access*, *8*, 142434–142445. https://doi.org/10.1109/ACCESS.2020.3010050

Caputo, A. C., & Pelagagge, P. M. (2008). Parametric and neural methods for cost estimation of process vessels. *International Journal of Production Economics*, *112*(2), 934–954. https://doi.org/10.1016/j.ijpe.2007.07.012

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, *39*(1), 1–22. https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

Dhara, S., & Delgado Barba, S. (2024). *Large language models in supply chain management* [Master's thesis]. Politecnico di Milano. https://hdl.handle.net/10589/222877

Elmousalami, H. H. (2021). Comparison of artificial intelligence techniques for project conceptual cost prediction: A case study and comparative analysis. *IEEE Transactions on Engineering Management*, *68*(1), 183–196. https://doi.org/10.1109/TEM.2020.2972078

Federal Reserve Bank of St. Louis. (2026). FRED: Federal reserve economic data. Retrieved June 2, 2026, from https://fred.stlouisfed.org/

GDELT Project. (2015). The GDELT global knowledge graph (GKG) data format codebook v2.1. Retrieved April 22, 2026, from http://data.gdeltproject.org/documentation/GDELT-Global_Knowledge_Graph_Codebook-V2.1.pdf

Google AI for Developers. (2026). Gemma 4 model card. https://ai.google.dev/gemma/docs/core/model_card_4

Gosling, J., & Naim, M. M. (2009). Engineer-to-order supply chain management: A literature review and research agenda. *International Journal of Production Economics*, *122*(2), 741–754. https://doi.org/10.1016/j.ijpe.2009.07.002

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. *Proceedings of the 34th International Conference on Machine Learning*, *70*, 1321–1330. https://proceedings.mlr.press/v70/guo17a.html

Koenker, R., & Bassett, G. (1978). Regression quantiles. *Econometrica*, *46*(1), 33–50. https://doi.org/10.2307/1913643

Lee, C.-Y., Chou, B.-J., & Huang, C.-F. (2022). Data science and reinforcement learning for price forecasting and raw material procurement in petrochemical industry. *Advanced Engineering Informatics*, *51*, 101443. https://doi.org/10.1016/j.aei.2021.101443

Leetaru, K. (2019). Comparing classical bag-of-word and neural sentiment algorithms. Retrieved April 22, 2026, from https://blog.gdeltproject.org/geg-comparing-classical-bag-of-word-and-neural-sentiment-algorithms/

Liu, Y., Yang, C., Huang, K., & Gui, W. (2020). Non-ferrous metals price forecasting based on variational mode decomposition and lstm network. *Knowledge-Based Systems*, *188*, 105006. https://doi.org/10.1016/j.knosys.2019.105006

London Metal Exchange. (2026). Market data. Retrieved June 2, 2026, from https://www.lme.com/en/Market-data

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

Luo, J., Zhang, W., Yuan, Y., Zhao, Y., Yang, J., Gu, Y., Wu, B., Chen, B., Qiao, Z., Long, Q., Tu, R., Luo, X., Ju, W., Xiao, Z., Wang, Y., Xiao, M., Liu, C., Yuan, J., Zhang, S., … Zhang, M. (2025). Large language model agent: A survey on methodology, applications and challenges. https://doi.org/10.48550/arXiv.2503.21460

Mandl, C., & Minner, S. (2023). Data-driven optimization for commodity procurement under price uncertainty. *Manufacturing & Service Operations Management*, *25*(2), 371–390. https://doi.org/10.1287/msom.2020.0890

Müller, A., & Grumbach, F. (2023). Predicting processing times in high mix low volume job shops. *16th International Doctoral Students Workshop on Logistics, Supply Chain and Production Management*. https://doi.org/10.25673/103491

Nederkoorn, J. B. M. (2017). *Innovating the quotation process at an eto company: A design-science approach* [Master's thesis]. Eindhoven University of Technology. https://research.tue.nl/en/studentTheses/innovating-the-quotation-process-at-an-eto-company-a-design-scien

Ozdemir, U. (2018). *Predicting social unrest using sentiment analysis* [Master's thesis]. Tilburg University. https://arno.uvt.nl/show.cgi?fid=147863

Qu, C., Dai, S., Wei, X., Cai, H., Wang, S., Yin, D., Xu, J., & Wen, J.-R. (2024). Tool learning with large language models: A survey. https://doi.org/10.48550/arXiv.2405.17935

Rokoss, A., Syberg, M., Tomidei, L., Hülsing, C., Deuse, J., & Schmidt, M. (2024). Case study on delivery time determination using a machine learning approach in small batch production

companies. *Journal of Intelligent Manufacturing*, *35*(8), 3937–3958. https://doi.org/10.1007/s10845-023-02290-2

Romano, Y., Patterson, E., & Candès, E. J. (2019). Conformalized quantile regression. *Advances in Neural Information Processing Systems*, *32*. https://doi.org/10.48550/arXiv.1905.03222

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, *36*. https://doi.org/10.48550/arXiv.2302.04761

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*(3), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Shen, S., Xia, L., Shuai, Y., & Gao, D. (2022). Measuring news media sentiment using big data for chinese stock markets. *Pacific-Basin Finance Journal*, *74*, 101810. https://doi.org/10.1016/j.pacfin.2022.101810

Simchi-Levi, D., Mellou, K., Menache, I., & Pathuri, J. (2025). Large language models for supply chain decisions. https://doi.org/10.48550/arXiv.2507.21502

Tambe, A., & Friedman, T. (2022). *Appendix b: GDELT dataset* (tech. rep.). Harvard Kennedy School Misinformation Review. https://misinforeview.hks.harvard.edu/wp-content/uploads/2022/01/tambe_appendix_b_20220114.pdf

United Nations General Assembly. (2015). Transforming our world: The 2030 agenda for sustainable development. https://sdgs.un.org/2030agenda

Verenich, I., Dumas, M., La Rosa, M., Maggi, F. M., & Teinemaa, I. (2019). Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. *ACM Transactions on Intelligent Systems and Technology*, *10*(4), 1–34. https://doi.org/10.1145/3331449

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.2210.03629

Appendix A

Manufacturing-Order Dataset Construction

This appendix details the construction of the manufacturing-order modelling dataset summarised in Section 4.3: the data-quality exclusion rules and the number of orders each removes, the 22-working-day scope restriction and the composition of the excluded tail, and the statistical distribution of the resulting modelling population.

A.1 Data-Quality Exclusion Rules

The quality step detects violations belonging to six categories derived from domain knowledge of the shop-floor production sequence and ERP recording behaviour. Two rules identify operation sequence inversions, that is, cases where a workstation type that must physically follow another is recorded as having started first: a laser operation appearing after cutting or bending operations, and a finishing operation preceding all others.

Three further rules address violations that are more consequential for duration accuracy. The most prevalent is the administrative closing delay: operators frequently complete physical production before attending to the formal system closure, causing the recorded end date to exceed the last real production session date by one or more days. Left uncorrected, this introduces a systematic upward bias in the target. A related type is ghost production after interruption, where the final quantity-producing session of a previously interrupted operation lasts under 60 seconds and is considered a system registration artefact rather than genuine activity. Finally, parent–child date inconsistencies arise when a child manufacturing order has a start or closing date that falls outside its parent order's window, indicating a hierarchy-level recording error.

A final rule flags orders whose last operation is both very short in duration and temporally isolated from all other operations by more than five days, suggesting a system close-out entry rather than real production activity. Orders flagged by any of these rules are excluded from the clean training population. Table A.1 reports the number of orders flagged by each rule. Because a single order may violate more than one rule, the per-rule counts are not mutually exclusive: the rules flag 3,582 distinct orders, of which 417 are flagged by two or more, and 3,478 fall within the completed-order modelling population and are removed, reducing it from 20,152 to 16,674 orders.

Table A.1: Manufacturing orders flagged by each data-quality rule on the full study population. A single order may trigger several rules, so the counts are not mutually exclusive (417 orders are flagged by two or more); 3,478 distinct orders are removed from the modelling population in total.

Rule	Orders flagged
Parent–child date inconsistency	1,689
Ghost production after interruption	1,273
Isolated short final operation	486
Administrative closing delay	472
Laser after cutting or bending	107
Finishing before remaining operations	4

A.2 Scope Restriction and the Excluded Tail

Orders whose effective duration exceeds 22 working days, approximately one calendar month, are excluded, and all reported duration metrics describe the resulting under-22-working-day population. This threshold is an operational, rather than a statistical, boundary. The duration estimator is designed as a short- to medium-horizon shop-floor predictor, and its feature set captures congestion, queue state, routing structure, and workstation throughput, all of which act over a near-term horizon. Orders extending beyond roughly one month are project-scale rather than shop-floor-scale: their realised duration is governed predominantly by procurement lead times, cross-order dependencies, and re-sequencing decisions that the ERP feature set does not observe, and that are instead addressed by the project-planning layer described in Section 4.4. The 22-working-day cut therefore marks the boundary between the duration model and the planning layer rather than the removal of inconvenient observations.

The excluded tail is also heterogeneous (Table A.2). It comprises 429 orders (2.57% of the cleaned population), of which the large majority (264) fall between 23 and 30 working days, durations that are entirely plausible for multi-stage assemblies, while only 30 exceed 50 working days and the raw tail extends implausibly to 144 working days. The near-threshold orders are excluded on the scope grounds above; the extreme upper tail, by contrast, is consistent with residual recording errors that the violation rules could not detect, such as orders whose incomplete production-session data prevents the closing-delay rule from firing. Because this exclusion removes the segment on which the model is weakest, the resulting metrics characterise the near-horizon population the model is intended to serve; the degradation on long-duration orders is examined directly in Section 5.2 and revisited as a limitation in Section 5.4.

Table A.2: Distribution of the cleaned orders excluded by the 22-working-day scope restriction ($n = 429$, 2.57% of the cleaned population).

Effective duration (working days)	Orders
23–30	264
31–50	135
51–100	26
> 100	4
Total (> 22)	429

A.3 Modelling-Population Distribution

After all exclusions the modelling population comprises 16,245 completed manufacturing orders. Table A.3 summarises the target variable (effective duration in working days), Table A.4 reports its composition by duration bracket, and Table A.5 characterises the structural complexity of the orders. The distribution is strongly right-skewed and zero-inflated: roughly two in five orders complete within a single working day, the median order is one working day, and the upper bound is the 22-working-day scope ceiling. This concentration at very short durations is the operational fact that motivates the two-stage architecture described in Section 4.3.

Table A.3: Summary statistics of the target variable (effective duration in working days) on the modelling population ($n = 16{,}245$).

Statistic	Value (working days)
Mean	3.29
Standard deviation	4.73
P10	0
P25	0
Median (P50)	1
P75	5
P90	10
P95	14
P99	20
Same-day (= 0 wd) share	41.2%

Table A.4: Composition of the modelling population by realised duration bracket ($n = 16{,}245$).

Duration bracket (working days)	Orders	Share
0 (same-day)	6,698	41.2%
1–2	3,419	21.0%
3–7	3,580	22.0%
8–14	1,747	10.8%
15–22	801	4.9%
Total	16,245	100.0%

Table A.5: Structural complexity of the modelling population: per-order distribution of operation, workstation, and component counts ($n = 16{,}245$).

Attribute	Mean	Median	P90	Max
Operations per order	2.36	2	4	7
Distinct workstations	2.31	2	4	7
Allocated components	2.70	1	6	245

Appendix B

Commodity Disruption Forecasting: Supplementary Materials

B.1 Alert Policy, Utility Function and Regression Sample Weighting

Alert policy parameters. The alert policy converts the sequence of weekly calibrated disruption probabilities p_t into discrete alert episodes. Rather than comparing p_t against a fixed cutoff, it maintains an *adaptive* probability threshold θ_t that rises when the model becomes confident and otherwise decays back toward a floor. Its tuned parameters are:

- **Probability threshold and floor.** An episode can start only when $p_t \geq \theta_t$, where the adaptive threshold θ_t is never allowed below a floor $\theta_{\min}$; this floor is what rejects persistently low-probability weeks.
- **Temporal decay rate ρ.** In weeks without a fresh high-confidence reading, the threshold relaxes multiplicatively, $\theta_t \leftarrow \theta_{t-1}\,(1-\rho\,\theta_{t-1})$, so that a stale, elevated threshold gradually returns toward $\theta_{\min}$ and the policy regains sensitivity.
- **Minimum probability delta δ.** A new episode additionally requires positive momentum, $\Delta p_t > \delta$, preventing alerts from triggering on a flat probability that merely drifts across the threshold.
- **High-probability override.** The momentum requirement is waived when $p_t \geq p_{\text{ovr}}$, so that an already-high disruption probability can fire even without a significant week-over-week increase.

A magnitude gate further requires the regression expected move to exceed a minimum size before an episode may start. These parameters are tuned jointly with the model hyperparameters in the Optuna search (Section 4.2).

Alert-utility function. The alert utility function is structured in two distinct layers, reflecting a conceptual separation between what constitutes a valid alert and how active alerts are penalised once triggered.

The first layer defines what an episode is worth in directional terms. For episode e, let A_e be its active weeks, partitioned into correct-direction weeks C_e and wrong-direction weeks W_e according to the sign of the realised weekly return r_t. Correct-direction capture and adverse variation are the symmetric sums

$$V_e^{+} = \sum_{t \in C_e} |r_t| \cdot 100, \qquad V_e^{-} = \sum_{t \in W_e} |r_t| \cdot 100,$$

expressed in percentage points of return. A *minimum capture gate* then zeroes the reward of any episode whose raw capture falls below $\kappa = 5$ percentage points, so that a negligible favourable move is not credited as a useful alert:

$$V_e^{+,\kappa} = V_e^{+} \cdot \mathbf{1}\left[V_e^{+} \geq \kappa\right].$$

The gate withholds reward only; it does not exempt the episode from penalties. A sub-threshold episode that moved against the alert has zero gated capture, so it cannot qualify as net useful and remains exposed to the alert-maintenance and adverse-week penalties; under the stricter re-score, it can also be charged its full adverse variation.

The second layer subtracts up to four penalty components, the first two of which depend on whether an episode is *net useful*. An episode is net useful when its gated capture is large relative to its adverse variation, $V_e^{+,\kappa}/V_e^{-} \geq \rho_{\mathrm{nu}} = 1.15$ (with the ratio taken as infinite when $V_e^{-} = 0$); intuitively, it captured substantially more than it gave back, so its adverse weeks are a pullback within a genuinely useful move rather than evidence of a wrong call. Let $\mathbf{1}_e^{\mathrm{nu}}$ denote this indicator.

The first component is the *operational penalty*, combining a per-active-week activity cost $c_{\mathrm{act}}|A_e|$, reflecting the burden of maintaining an active procurement alert, with a flat per-wrong-week cost $c_{\mathrm{wk}}|W_e|$ that is scaled down by a factor $s_{\mathrm{mix}} = 0.35$ for net-useful episodes, so that they are not doubly penalised for transient adverse weeks. For a net-useful episode the adverse variation V_e^{-} is folded into this bundle; for an episode that is not net useful it is kept outside this bounded component, allowing the stricter re-score below to charge it separately. The bundle is then subject to a *penalty ceiling*, capped at a fraction $\eta = 0.50$ of the gated capture:

$$\Pi_e^{\mathrm{op}} = c_{\mathrm{act}}|A_e| + c_{\mathrm{wk}}^{\mathrm{adj}}|W_e| + \mathbf{1}_e^{\mathrm{nu}} V_e^{-}, \qquad \Pi_e^{\mathrm{op},*} = \min\left(\Pi_e^{\mathrm{op}},\ \eta \cdot V_e^{+,\kappa}\right).$$

The ceiling prevents the pullback within a good move, together with flat-rate costs, from driving moderate episodes deeply negative.

The second component is an optional *strict adverse-variation charge* for episodes that are not net useful, $(1 - \mathbf{1}_e^{\mathrm{nu}})V_e^{-}$, charged in full and *outside* the ceiling. This term is not used to train or select the reference policy, but is useful as a conservative post-hoc stress test: it asks how the same fixed alerts would score if a non-materialising alert were charged in proportion to the realised move against it. A net-useful episode never triggers this term, so a genuinely useful but volatile alert remains protected by the ceiling.

The third component is a *re-fire penalty* $\phi_e = \phi_e^{\mathrm{spam}} + \phi_e^{\mathrm{time}}$ discouraging redundant intra-episode re-fires. Letting $\mathscr{R}_e$ denote the raw-alert weeks beyond the initial trigger and $V_{e,<k}^{+}$ the correct-direction variation already captured before re-fire k, the *spam* term

$$\phi_e^{\mathrm{spam}} = c_{\mathrm{rf}} \sum_{k \in \mathscr{R}_e} V_{e,<k}^{+}, \qquad c_{\mathrm{rf}} = 0.10,$$

charges re-fires that occur after most of the move has played out most heavily. The *timing* term charges up to $\psi = 2.0$ percentage points per re-fire that is not followed by continued favourable movement, measured as the directional return over the next $h_{\mathrm{rf}} = 2$ weeks relative to a minimum continuation $\delta_{\mathrm{c}} = 2.0$ points; a re-fire near a local extremum, where the move does not continue, thus incurs the full timing penalty.

The fourth component is a *stability penalty* σ_e targeting alerts that are quickly contradicted. Let s_e be the number of weeks the episode survives before an opposite-direction episode begins, capped at the horizon $H = 8$ weeks. An episode truncated early incurs

$$\sigma_e = c_{\mathrm{stab}} \frac{H - s_e}{H} \quad \text{for } s_e < H,$$

and zero otherwise, so that a directional call reversed shortly after firing is penalised in proportion to how early the reversal occurs.

The reference utility for direction d sums the gated capture net of the operational, re-fire, and stability components:

$$U_d^{\text{ref}} = \sum_e \left[V_e^{+,\kappa} - \Pi_e^{\text{op},*} - \phi_e - \sigma_e \right], \tag{B.1}$$

and the per-direction values sum to $U_{\text{total}}^{\text{ref}} = U_{\text{down}}^{\text{ref}} + U_{\text{up}}^{\text{ref}}$. The stricter ex-post utility used only for the robustness re-score is

$$U_d^{\text{strict}} = U_d^{\text{ref}} - \sum_e (1 - \mathbf{1}_e^{\text{nu}}) V_e^{-}.$$

The structural constants κ, c_{rf}, η, ψ, h_{rf}, δ_{c}, and the net-useful relief parameters ρ_{nu} and s_{mix} are fixed by design and excluded from optimisation, since tuning them would risk the optimiser gaming the quality definition itself, for instance by weakening the re-fire penalty in configurations that re-fire near local extrema. The cost parameters c_{act} and c_{wk} and the stability penalty c_{stab} (searched over $[0.5, 8.0]$) are selected jointly with the model hyperparameters in the Optuna search.

Reference versus strict ex-post utility. The reference utility U^{ref} is both the model-selection objective and the headline reporting metric in Chapter 5. This keeps the reported reference system aligned with the objective it was actually optimised to satisfy. The stricter U^{strict} score is deliberately separated from model selection. It is a sound way to stress-test a fixed deployed alert policy, since a directional alert that resolves the wrong way is operationally worse than silence, but it is a poor direct optimisation target for two reasons. First, it encodes a specific cost preference, namely how a wrong alert is valued relative to a missed one, and that preference is buyer-specific (Section 5.1). Second, optimised directly, the charge is most cheaply reduced not by improving alert timing but by suppressing alerts altogether, driving the policy toward a degenerate over-conservative regime that captures few genuine disruptions. The stricter score is therefore reported only as a fixed-policy robustness diagnostic in Appendix B.5.

Regression sample weighting. Regression training applies a two-component sample weighting scheme. A relevance weight sharply downweights stable weeks while upweighting those whose realised directional return exceeds the disruption threshold. A magnitude weight further amplifies training on larger moves via:

$$w_{\text{mag}} = 1 + \alpha z^{\gamma}$$

where $z \in [0, 1]$ normalises move size, $\alpha > 0$ controls the strength of the magnitude boost, and $\gamma > 0$ controls its curvature. The final sample weight is the product of base sample weights, relevance, and magnitude components, normalised to mean 1 within each fold. The weighting strength α is selected jointly with model hyperparameters during the Optuna search; γ is held fixed as a design choice.

B.2 Latent-Regime Feature Construction

Model specification. For each training fold, the regime detector is defined as a Gaussian Mixture Model with $K = 4$ components and diagonal covariance matrices. The GMM is fitted on a standardised matrix of market-state features using zero-mean, unit-variance scaling estimated from the fold-specific fitting sample. The implementation uses five random initialisations, a covariance regularisation term of 5×10^{-4} and a maximum of 300 expectation-maximisation (EM) iterations. Convergence is declared when the change in log-likelihood between successive EM

iterations falls below the solver's default tolerance (Dempster et al., 1977); if this criterion is not met within the iteration budget, the fit is discarded. If fitting fails or the model does not converge, the regime module falls back to a neutral signal by assigning uniform membership probabilities of $1/K$ to all four regimes. This ensures that downstream classification remains well-defined even when a reliable regime partition cannot be estimated for a given fold.

Feature design selection. The production pipeline does not perform an unrestricted combinatorial search over all feature subsets at runtime. Instead, it evaluates a small predefined set of candidate feature combinations and selects the best available configuration for the current training fold. The broader candidate pool includes GDELT-derived latent factors, speculative positioning variables, inventory level and inventory change measures, the LME 3-month spread, term-structure indicators, the nickel-to-copper price ratio, the price levels of other metals such as aluminium and gold, realised volatility at 8-week and 26-week horizons, and the VIX.

Each candidate design is evaluated using two complementary criteria. First, the pipeline computes a restricted downstream predictive score by fitting a lightweight XGBoost classifier on the regime probabilities alone and evaluating its PR-AUC for both the downside and upside 8-week disruption labels over a small set of post-July 2021 walk-forward validation starts. Second, it computes regime-health diagnostics designed to rule out degenerate clustering behaviour. These constraints require: (i) median maximum posterior probability no greater than 0.90, to avoid near-hard assignments; (ii) mean normalised Shannon entropy (Shannon, 1948) of the posterior distribution,

$$\bar{H} = -\frac{1}{\log K} \sum_{k=1}^{K} p_k \log p_k \in [0,1],$$

of at least 0.20, to preserve genuinely soft memberships; (iii) minimum cluster share of at least 0.05, to prevent regime collapse; and (iv) pre-2019 dominant regime concentration no greater than 0.90, to avoid constructing a single historical catch-all cluster for the pre-transition era.

Selection is performed by ranking candidate designs in the following order: health-constrained solutions are preferred to invalid ones; among valid designs, higher composite design scores are preferred; remaining ties are broken by lower median maximum posterior probability and then by higher normalised entropy. The composite design score combines the restricted downstream PR-AUC with entropy- and concentration-based penalties, so the selected regime design must be both predictively useful and structurally well-behaved.

Causal construction and posterior softening. To preserve causal validity, the GMM is estimated separately within each walk-forward fold using only data available prior to the fold boundary, so that regime assignments carry no look-ahead information. The fitting sample is restricted to observations from 1 January 2019 onwards, reflecting the structural reorganisation of nickel market dynamics during the late 2010s. This period was shaped by the emergence of battery-grade nickel demand and the associated realignment of inventory flows and trade patterns, which rendered pre-transition market regimes less representative of the current price formation environment.

Raw GMM posterior probabilities can still be overly concentrated, even when the underlying clustering solution is acceptable. To preserve a graded notion of market state rather than collapsing to an almost categorical label, the posterior vector is softened using a temperature transform (Guo et al., 2017) and then blended with the fitted mixture prior. Let $p_{t,k}$ denote the raw posterior probability for regime k in week t, and let π_k denote the fitted mixture weight. The pipeline applies

$$\tilde{p}_{t,k} \propto p_{t,k}^{1/T}, \qquad \hat{p}_{t,k} = (1-\alpha)\,\tilde{p}_{t,k} + \alpha\,\pi_k,$$

with temperature $T = 1.8$ and prior-blending coefficient $\alpha = 0.10$, followed by renormalisation. This transformation flattens excessively sharp posterior mass while retaining relative regime ordering and allowing the prior cluster weights to stabilise the representation. These values are fixed by design rather than tuned; the sensitivity of the regime block to T, α, and the post-transition anchor year is examined in Appendix B.8.

Integration with the downstream classifier. The final output of the regime module is a four-dimensional vector of softened regime probabilities, one probability per component. These variables are appended to the main weekly feature matrix before XGBoost training; they do not replace the original predictors, but provide a compact contextual layer through which the downstream model can learn regime-conditional associations. To preserve consistency between training-time construction and later evaluation or inference, the fitted fold-specific GMM parameters, scaler, and selected input columns are serialised as replay artifacts. This allows the pipeline to reconstruct the exact fold-local regime probabilities associated with each validation window, rather than approximating them with a single globally fitted model.

B.3 Detailed Evaluation Design and Metric Definitions

All performance figures are derived from OOF predictions produced by the walk-forward cross-validation scheme described in Section 4.2, so that no evaluation observation was available to any model at the time of training. The evaluation window begins on 01 July 2021 and ends on 22 May 2026 for both metals. The copper OOF disruption probabilities used as cross-asset features in the nickel model are restricted to the same fold-safe subset. The final eight weeks of each sample carry undefined labels and are excluded throughout.

Three prediction surfaces are evaluated for each (metal, direction) pair. The **Round 2 classifiers** (Section 4.2) produce the OOF disruption probabilities that drive the pipeline. The **regression models** estimate the expected maximum directional return over the eight-week horizon anchored at each prediction week, as defined in Section 4.2. The **alert policy** converts classifier probabilities into procurement alert episodes via the threshold logic optimised using Equation B.1.

A no-skill baseline that assigns a constant probability equal to the empirical label prevalence of each task is used as the lower bound for classifier evaluation. In the final OOF samples, the positive-class prevalence is 0.190 for copper-down, 0.298 for copper-up, 0.206 for nickel-down, and 0.254 for nickel-up.

Classification. The primary performance criterion for the pipeline is alert utility (Section 5.1), since the classifiers are optimised against the reference utility in Equation B.1 rather than any classification surrogate. PR-AUC is reported as the primary classifier-specific diagnostic: it assesses discriminative ability in isolation, independently of the alert policy, and is preferred over receiver operating characteristic area under the curve (ROC-AUC) because the positive class is a minority at roughly 19–30% across the four tasks. ROC-AUC is included as a complementary rank-order measure. Precision, recall, and F_1 are evaluated at the operating threshold selected by the alert utility optimisation, providing an operating-point summary alongside the threshold-free curve metrics. The Brier score provides a joint measure of sharpness and calibration.

Alert utility and episode metrics. A positive U_d indicates positive net utility under the reference scoring rule after deducting the penalties defined in Appendix B.1. The stricter post-hoc version is denoted U_d^{strict} where it is used. The following episode-level quantities are also reported: episode

count, mean episode duration (weeks), directional accuracy (fraction of episodes in which the price moved in the alerted direction by at least 5%), and mean gated correct-direction variation per episode ($\bar{V}_e^+$, pp).

Regression and confidence interval metrics. The regressor is gated behind the classifier in deployment and is never called on non-alert weeks. Evaluating it on the full OOF sample would therefore measure a situation that never occurs in practice and would inflate errors given that the model's sample weighting scheme deliberately down-weights non-disruption weeks during training. Two restricted populations are used instead.

The *primary population* comprises all weeks where the classifier fires an alert, including re-fires within episodes. Each firing week t is scored against its own realised maximum directional return over $[t, t+8]$, which is the horizon the regressor was trained to predict. The number of distinct episodes represented among these weeks is reported as a transparency note.

The *diagnostic population* comprises all weeks where the true disruption label is positive, independent of the alert policy. This answers a complementary question: conditional on a real disruption occurring, how well does the regressor estimate its magnitude.

For both populations, point-estimate performance is reported as MAE (percentage return units), R^2, and Bias (mean signed error; positive values indicate systematic under-prediction of move magnitude). Confidence interval performance is reported as coverage (fraction of weeks where $r_{\text{true}} \in [\text{ci\_lower}, \text{ci\_upper}]$, target 0.80) and mean interval width. Both confidence interval (CI) metrics use the same population filters as the point-estimate metrics.

Calibration. Classifier calibration is assessed via the expected calibration error (ECE), computed over ten equal-width probability bins as the bin-size-weighted mean of $|\bar{p}_b - \bar{y}_b|$. The Brier score, already reported under classification, provides the companion joint metric. For a constant-probability classifier, the corresponding reference Brier score is $p(1-p)$ for the task-specific prevalence p; in the present results those reference values range from approximately 0.154 to 0.209. A Brier score below this reference confirms that a classifier adds probabilistic value beyond the base rate.

B.4 Alert-Utility Robustness Checks

Because the reference alert policies generate a small number of episodes, the headline utility figures are complemented by two lightweight robustness checks. The first is a leave-one-episode-out diagnostic: each realised alert episode is removed in turn and the remaining total utility is recomputed from the episode-level contribution table. This does not create a formal confidence interval; instead, it tests whether the positive result is dominated by a single episode. Table B.1 summarises the resulting range for the reference policies.

Table B.1: Leave-one-episode-out robustness of reference alert utility. U_{total} is reported in percentage points of price return.

Metal	Episodes tested	Reference U_{total}	Min. U_{total}	Max. U_{total}
Copper	17	94.1	79.7	95.9
Nickel	16	200.3	157.6	201.8

In both metals, the total utility remains positive under every single-episode removal, with the most influential episode reducing the total by at most 15.3% (copper) and 21.3% (nickel), indicating that the reference-policy result is not driven by one isolated alert episode.

The second check varies the cost side of the alert-utility rule while keeping the reference alert episodes fixed. The sensitivity grid scales the active-week penalty, wrong-direction-week penalty, re-fire penalty coefficient, and episode-stability penalty jointly and one at a time. Table B.2 reports the range across the tested settings. The joint cost multiplier is evaluated at $0.5, 0.75, 1.25, 1.5$, and 2.0, and each cost component is also perturbed individually at $0.5, 1.5$, and 2.0, with the baseline setting also included. Utility remains positive for both metals even under the most punitive tested setting, where all four cost terms are doubled.

Table B.2: Cost-sensitivity range for reference alert utility. The grid contains 18 settings per metal, including the baseline scoring rule and one-at-a-time or joint cost multipliers.

Metal	**Settings tested**	**Reference U_{total}**	**Min. U_{total}**	**Max. U_{total}**
Copper	18	94.1	63.8	112.4
Nickel	18	200.3	156.7	227.6

The sensitivity range is wider for copper, with utility falling by at most 32.2% (copper) and 21.7% (nickel) across the grid, but both metals remain positive, so the qualitative conclusion does not depend on the exact baseline cost coefficients.

B.5 Stricter Fixed-Policy Re-Score

The reference results above use the utility rule under which the alert policy was selected. As an additional conservative diagnostic, the same fixed out-of-fold alert episodes are re-scored under U^{strict}, which adds the full adverse-variation charge for episodes that are not net useful. No model is retrained, no alert thresholds are re-optimised, and no new episodes are generated. The exercise therefore answers a narrower question: how much of the reference utility survives if the realised alerts are judged under a more punitive post-hoc cost convention.

Table B.3: Strict ex-post re-score of the reference alert policies. Both utilities are in percentage points of price return and are computed on the same fixed alert episodes.

Metal	**Dir.**	U_d^{ref}	U_d^{strict}	**Change**
Copper	Down	32.5	26.0	-6.4
Copper	Up	61.6	42.4	-19.2
Nickel	Down	111.5	66.7	-44.8
Nickel	Up	88.8	66.7	-22.1

Under this stricter re-score, total utility remains positive for both metals, falling from 94.1 to 68.4 percentage points for copper and from 200.3 to 133.4 percentage points for nickel. The result is therefore directionally reassuring, but it is not the primary reference score because the policy was not trained to optimise this stricter objective.

The stricter ablation re-score preserves the main nickel conclusion: all substantive feature families improve the full reference model. Copper is more sensitive to the scoring rule. Under the reference utility, both GDELT and Chronos pruning improve copper; under the strict re-score, only GDELT pruning remains favourable. This is best read as evidence that copper's marginal

Table B.4: Strict ex-post re-score of commodity ablation totals. ΔU^{strict} is relative to the strict re-score of each metal's full reference model.

Metal	Variant	$U^{\text{strict}}_{\text{total}}$	ΔU^{strict}
Copper	Baseline	68.4	0.0
Copper	Minus GDELT	81.8	+13.4
Copper	Minus Chronos	65.5	-3.0
Copper	Minus regime prob.	27.9	-40.6
Nickel	Baseline	133.4	0.0
Nickel	Minus GDELT	67.8	-65.7
Nickel	Minus Chronos	105.2	-28.2
Nickel	Minus regime prob.	99.7	-33.7
Nickel	Minus copper transfer	114.8	-18.7

feature conclusions are more cost-convention-sensitive than nickel's, not as a replacement for the reference-policy ablation table in the main chapter.

B.6 Classification and Calibration Diagnostics

The classification and calibration diagnostics remain important, but they are supporting evidence rather than the main story. All four classifiers beat their prevalence-only baselines, so the models are learning genuine ranking structure. The operating thresholds selected by alert-utility optimisation are also intentionally conservative, which helps explain why the policies can show moderate precision with relatively low recall. This is consistent with a procurement setting in which false persistence and wrong-direction alerts carry real cost.

Table B.5: OOF classification performance for the copper and nickel directional disruption classifiers. Precision, recall, and F_1 are evaluated at the operating thresholds selected by the alert utility optimisation (Section 4.2).

Metal	Dir.	PR-AUC	ROC-AUC	Prec.	Rec.	F_1	Brier
Copper	Down	0.426	0.632	0.643	0.191	0.295	0.160
Copper	Up	0.539	0.705	0.706	0.324	0.444	0.234
Nickel	Down	0.522	0.790	0.619	0.255	0.361	0.143
Nickel	Up	0.409	0.680	0.577	0.238	0.337	0.189

Within that supporting role, two points matter most. The first is that the classifiers have enough ranking power to justify the disruption framing itself. The second is that those same metrics do not tell us which feature families deserve to remain. A variant can improve total utility while moving one directional PR-AUC slightly up and another only marginally, or it can preserve classifier metrics while changing the policy economics materially.

The same supporting interpretation applies to calibration. The reported probabilities are useful for ranking and alert gating, but they are not strong enough to be treated as fully self-interpreting risk numbers. This is especially important because the agent surfaces probabilities directly to decision-makers: calibration that is merely acceptable still calls for contextual explanation rather than raw numerical display alone.

The key calibration conclusion is therefore modest and metal-dependent. As the Δ column shows, the nickel classifiers add probabilistic information beyond the base rate, nickel-down falls clearly below its reference and nickel-up sits essentially on it, together with the lowest ECE values

Table B.6: Classifier calibration metrics on the full OOF sample. The reference Brier is the prevalence-only value $p(1-p)$, i.e. the score obtained by always predicting the base rate; a model Brier below this reference ($\Delta < 0$) indicates probabilistic information beyond the base rate. ECE is the expected calibration error over ten equal-width probability bins.

Metal	**Dir.**	**Brier score**	**Ref. Brier $p(1-p)$**	**Δ (Brier $-$ ref.)**	**ECE**
Copper	Down	0.160	0.154	$+0.006$	0.130
Copper	Up	0.234	0.209	$+0.025$	0.187
Nickel	Down	0.143	0.163	-0.020	0.109
Nickel	Up	0.189	0.189	0.000	0.097

in the table (0.097 to 0.109). The copper classifiers, by contrast, sit at or marginally above their Brier references ($\Delta > 0$), so their scores are best treated as ranking signals rather than as well-calibrated probabilities.

This separation is what reconciles the calibration picture with the headline result. The pipeline scores well on alert utility (Section 5.1) because that metric depends on correctly ranking and timing disruptions, not on the probabilities being literally accurate: a classifier can order disruption weeks above calm weeks, and so drive useful alert actions, while still attaching numerically inaccurate probabilities to them. Even so, the calibration picture should not be overstated: only nickel improves on the base rate at all (and only on the downside, since nickel-up merely matches it), and with ECE near 0.10 even nickel is at best moderately calibrated. Copper does not beat the base rate, so its probability outputs should be treated as a relative risk signal rather than as the true probability of the event. This is the practical reason the agent surfaces probabilities with contextual framing rather than as standalone odds, and why the evaluation is anchored on alert utility throughout.

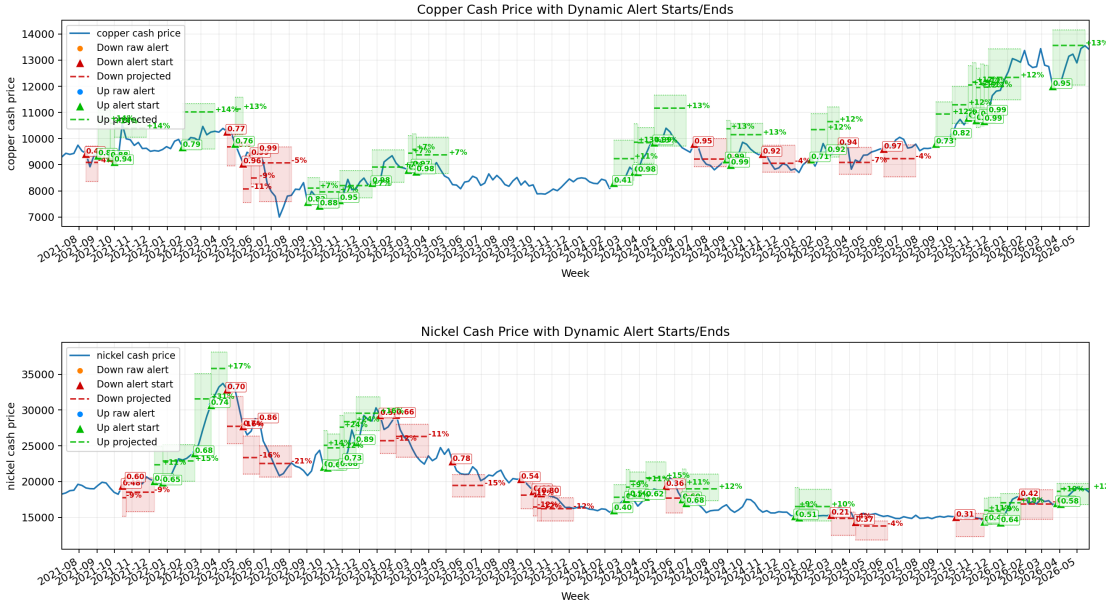

Figure B.1: OOF alert episodes for copper (top) and nickel (bottom) overlaid on the weekly price series. Downside windows are shaded red; upside windows are shaded blue. Disruption threshold levels $\pm\tau$ are indicated.

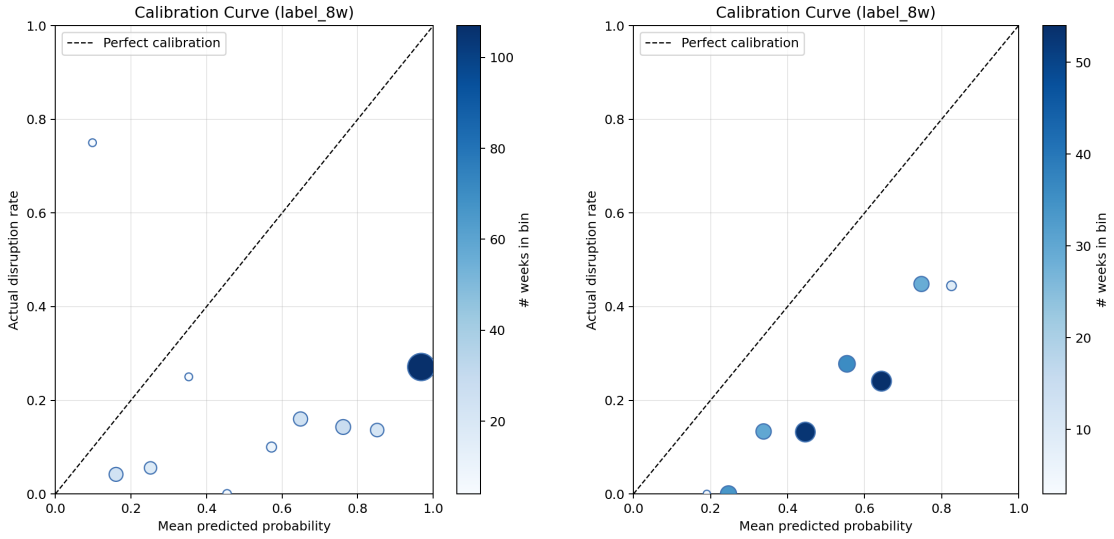

Figure B.2: Reliability diagrams for the copper and nickel directional disruption classifiers (OOF). Left: copper. Right: nickel. Observed disruption frequency is plotted against mean predicted probability within equal-width bins. The diagonal indicates perfect calibration; bar widths are proportional to bin population.

B.7 Latent Regime Diagnostics

The main chapter uses the regime module only through its contribution to out-of-fold alert utility and related diagnostics. The figures in this section expose the fitted latent-state behaviour more directly. For exposition, the latent components are assigned cautious post-hoc economic shorthand labels using three pieces of evidence exported by the pipeline: (i) the empirical regime profile tables, which report mean contemporaneous and 8-week forward returns and directional disruption rates by dominant regime; (ii) the standardised centroid tables, which show which regime-input features are relatively high or low within each component; and (iii) the temporal-coherence and transition diagnostics below. These labels should therefore be read as interpretive summaries of the replayed latent states, not as intrinsic or fold-invariant economic categories.

Table B.7: Empirical replayed profile of the copper latent regimes. Cluster weight is the share of weeks assigned to each dominant regime in the replayed evaluation sample. Forward return is the mean 8-week ahead return conditional on the dominant regime.

Regime	Weight	Mean wk. ret. (%)	Mean 8w fwd ret. (%)	Down rate	Up rate
R0	0.090	−0.198	−2.578	0.421	0.211
R1	0.421	0.358	2.735	0.082	0.336
R2	0.273	−0.135	−1.273	0.342	0.301
R3	0.216	0.195	2.684	0.158	0.333

Copper: Assigned Labels

R0: Downside inventory-stress regime. This regime combines the most negative 8-week forward return (−2.58%) with the highest downside disruption rate (0.421). Its centroid also shows above-average inventory pressure and elevated short-horizon realised volatility, so it is interpreted as a downside state in which supply-demand imbalance and unstable price conditions coincide.

R1: Constructive low-volatility regime. R1 has the strongest positive 8-week forward return (+2.73%) and the lowest downside disruption rate (0.082). At the same time, its centroid indicates below-average inventory pressure and relatively subdued realised volatility, which justifies interpreting it as the calmest and most constructive copper state.

R2: Transitional mixed regime. R2 remains weaker than R1 in both contemporaneous and forward returns, but it is not as adverse as R0. Its upside disruption incidence remains material (0.301), while the centroid does not show the strongest stress signature on inventory or volatility. This makes it more plausibly a mixed or transitional environment than a clearly bearish or clearly bullish one.

R3: Tight/bullish high-price regime. R3 has a positive 8-week forward return (+2.68%), the highest aluminium-price centroid, and the strongest LME spread reading among the copper regimes. Those features point to a firmer and tighter pricing environment. Its realised volatility remains above the very calm R1 state, so the label emphasises tightness and elevated price conditions rather than simple low-risk bullishness.

Table B.8: Empirical replayed profile of the nickel latent regimes. Cluster weight is the share of weeks assigned to each dominant regime in the replayed evaluation sample. Forward return is the mean 8-week ahead return conditional on the dominant regime.

Regime	Weight	Mean wk. ret. (%)	Mean 8w fwd ret. (%)	Down rate	Up rate
R0	0.164	−0.180	−3.871	0.366	0.244
R1	0.428	0.538	0.807	0.146	0.268
R2	0.192	0.194	3.725	0.157	0.314
R3	0.215	−1.296	−2.099	0.268	0.196

Nickel: Assigned Labels

R0: Bearish inventory-overhang regime. R0 records a negative weekly return and the most negative 8-week forward return (−3.87%), together with the highest downside disruption rate (0.366). Its centroid also shows above-average inventory pressure, which supports the interpretation of a bearish state associated with inventory overhang rather than only short-lived volatility.

R1: Constructive/speculative-support regime. R1 has the highest weekly return (+0.54%), a positive forward return, and the lowest downside disruption rate (0.146). The centroid also shows an above-average speculative positioning signal, measured through the normalised net imbalance between long and short LME investment-fund positions. This combination motivates the label of a constructive regime supported by speculative positioning.

R2: Tight-market upside regime. R2 delivers the strongest 8-week forward return (+3.73%) and the highest upside disruption rate (0.314). Its centroid simultaneously shows below-average inventory pressure, which is consistent with a relatively tighter physical market backdrop and supports an upside-oriented label.

R3: High-volatility stress regime. R3 has the most negative contemporaneous weekly return (−1.30%) and the highest realised volatility centroid, while still showing a materially elevated

downside disruption rate (0.268). Relative to R0, this looks less like inventory overhang and more like an acute stress state dominated by turbulent price action.

Figure B.3 plots the temporal coherence of the inferred regimes for copper and nickel. In both metals the dominant state evolves in multi-week blocks rather than alternating erratically from one week to the next, which is consistent with the goal of encoding slowly changing market context instead of short-lived noise.

Figure B.4 reports the corresponding empirical transition matrices. The diagonal persistence is material but not overwhelming, and off-diagonal mass remains visible across several transitions, indicating that the regime process is neither collapsed into a single dominant state nor trapped in near-absorbing clusters. This supports the interpretation of the regime probabilities as soft contextual features rather than as a hard partition of the price history.

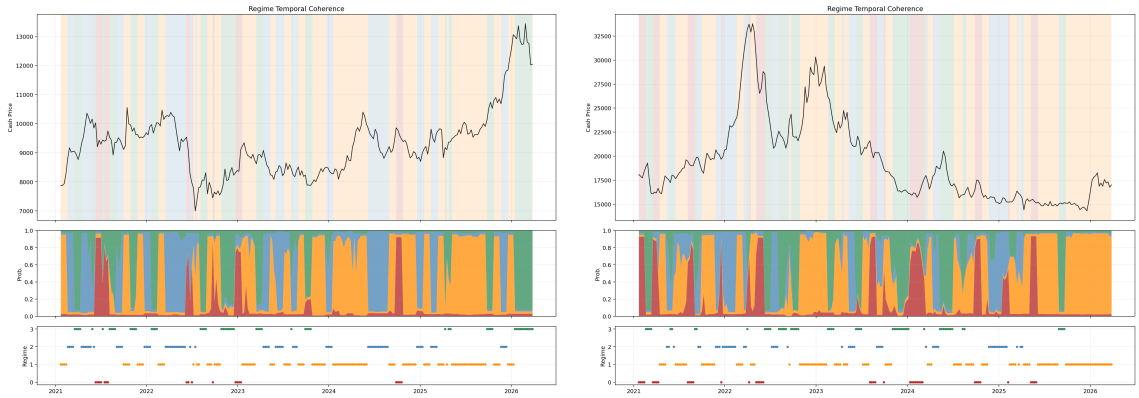

Figure B.3: Temporal coherence diagnostics for the fold-replayed latent regime assignments. Left: copper. Right: nickel. The plots show whether inferred dominant states persist in coherent blocks over time rather than flipping week to week.

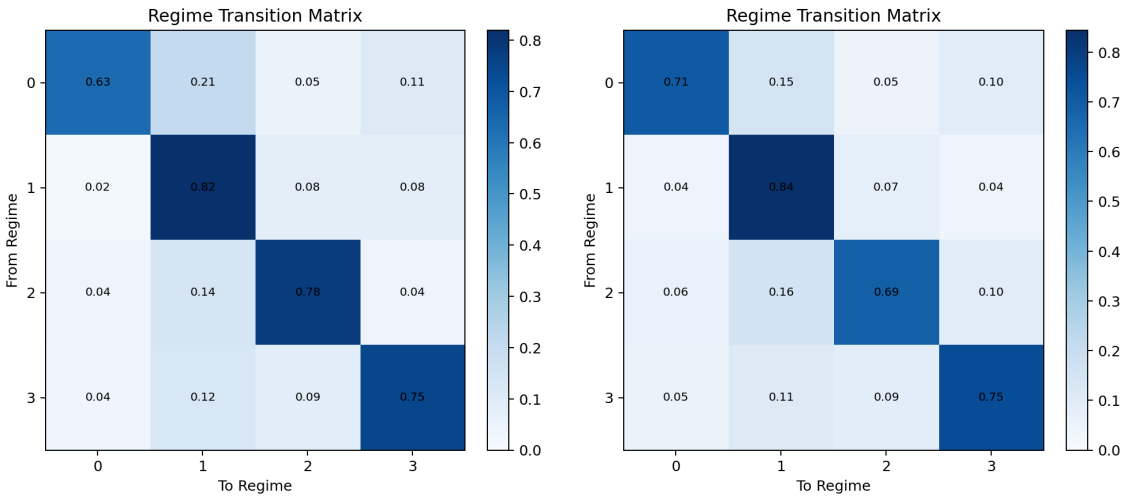

Figure B.4: Empirical transition matrices for the fold-replayed latent regime assignments. Left: copper. Right: nickel. Diagonal mass reflects state persistence, while off-diagonal mass shows that transitions remain possible and the regime process is not degenerate.

B.8 Regime Block Sensitivity Analysis

The latent-regime block introduces three hyperparameters that are fixed in the production pipeline (Section 4.2, Chapter 4): the anchor year that restricts the fitting sample to the modern post-transition era, the posterior-softening temperature T, and the prior-blending coefficient α. This section reports a sensitivity analysis that examines how the regime block responds to perturbations of each of these three knobs, for both copper and nickel.

Protocol. For each commodity, the feature set is fixed to the design selected by the production pipeline at the baseline configuration (anchor $= 2019$, $T = 1.8$, $\alpha = 0.10$). Holding this feature set constant, each of the three hyperparameters is then swept independently, with the other two pinned to their baseline values. The swept grids are anchor $\in \{2016, 2017, 2018, 2019, 2020, 2021\}$, $T \in \{1.0, 1.2, 1.5, 1.8, 2.2, 2.6\}$ and $\alpha \in \{0.0, 0.05, 0.10, 0.20, 0.30\}$. To match the production training pipeline, all fits use weekly data from 1 January 2015 onwards.

For each configuration, a walk-forward evaluation is performed that mirrors the production pipeline's leakage-safe protocol: at every fold boundary (spaced thirteen weeks apart, beginning January 2022), the GMM is refit on training data prior to that boundary, a lightweight XGBoost classifier is trained on the resulting softened regime probabilities, and held-out predictions are accumulated over a fifty-two-week validation window. The reported *regime-only PR-AUC* is the average of pooled downside and upside PR-AUC across folds, and is therefore directly comparable to the headline regime-only figures produced by the production pipeline's evaluation step.

Four *health gates* are also recorded: median maximum posterior probability, mean normalised Shannon entropy, minimum cluster share, and pre-2019 dominant-regime concentration. As in the production feature-set selector, health gates are evaluated on a single in-sample fit over the full anchored window rather than on per-fold replays, since the pre-2019 concentration metric is structurally zero when restricted to the post-2022 evaluation period. A configuration is reported as *valid* if it passes all four health gates simultaneously.

Anchor year. Table B.9 reports the marginal sweep over anchor year, with T and α pinned at baseline. For copper, only the 2019 anchor satisfies the health gates: every other tested year is rejected, with anchors ≤ 2018 failing the pre-2019 concentration gate (a single dominant regime would absorb the entire pre-anchor period) and anchors ≥ 2020 failing the median maximum posterior probability gate (the representation collapses to near-categorical assignments).

For nickel, four of the six anchor years are valid (2017–2020), and the regime-only PR-AUC varies between 0.208 and 0.270 across them, with the 2019 anchor producing the highest valid score (0.270). The two invalid anchors (2016 at 0.211 and 2021 at 0.192) sit below the valid optimum as well, so 2019 is the best-performing configuration across the entire nickel anchor grid. A secondary factor is that later anchor years reduce the number of fitting rows (from 533 at 2016 to 272 at 2021), so part of the deterioration at the 2021 anchor reflects a smaller fitting sample rather than a worse structural choice.

Taken together, the empirical optimum for both commodities coincides with the 2019 anchor selected on structural grounds in the main text.

Posterior-softening temperature. Table B.10 reports the sweep over the temperature T. For both commodities, the lowest tested temperatures ($T = 1.0$ and $T = 1.2$) fail the entropy gate: without sufficient softening, the posterior collapses too sharply onto a single component and the representation becomes effectively categorical. Within the valid range $T \in \{1.5, 1.8, 2.2, 2.6\}$, the regime-only PR-AUC varies by less than 0.020 for both commodities (0.265–0.270 for nickel,

Table B.9: Marginal sweep over the post-transition anchor year, with $T = 1.8$ and $\alpha = 0.10$ fixed at the baseline. *Fit rows* is the number of weekly observations used to fit the GMM. *Pre-2019 conc.* is the fraction of weeks in $[2015, \text{anchor})$ assigned to a single dominant regime; the health gate requires this value to be at most 0.90. *Valid* indicates whether all four health gates are satisfied. PR-AUC values are produced by walk-forward refit/score, pooled across folds, and directly comparable to the production pipeline's headline regime-only figures.

Anchor	Fit rows	PR-AUC	Med. max prob	Norm. entropy	Min cluster share	Pre-2019 conc.	Valid
Nickel							
2016	533	0.211	0.863	0.405	0.129	0.923	no
2017	480	0.213	0.827	0.465	0.084	0.686	yes
2018	428	0.208	0.865	0.432	0.081	0.459	yes
2019	376	**0.270**	0.873	0.424	0.092	0.569	yes
2020	324	0.247	0.773	0.499	0.136	0.766	yes
2021	272	0.192	0.913	0.344	0.045	0.527	no
Copper							
2016	533	0.228	0.921	0.332	0.089	1.000	no
2017	480	0.214	0.911	0.331	0.089	1.000	no
2018	428	0.265	0.898	0.360	0.092	1.000	no
2019	376	**0.260**	0.836	0.394	0.092	0.823	yes
2020	324	0.216	0.917	0.298	0.089	0.877	no
2021	272	0.234	0.921	0.298	0.077	0.856	no

0.252–0.271 for copper). The baseline $T = 1.8$ lies inside this stable plateau in both cases, indicating that the chosen temperature is not in a sharp optimum and that small perturbations have negligible impact on downstream predictive utility.

Table B.10: Marginal sweep over the posterior-softening temperature T, with the anchor year fixed at 2019 and $\alpha = 0.10$. Columns as in Table B.9.

T	PR-AUC	Med. max prob	Norm. entropy	Min cluster share	Pre-2019 conc.	Valid
Nickel						
1.0	0.276	0.919	0.324	0.092	0.574	no
1.2	0.263	0.911	0.349	0.092	0.569	no
1.5	0.265	0.898	0.386	0.092	0.569	yes
1.8	**0.270**	0.873	0.424	0.092	0.569	yes
2.2	**0.270**	0.836	0.471	0.089	0.569	yes
2.6	0.266	0.799	0.516	0.086	0.569	yes
Copper						
1.0	0.261	0.914	0.305	0.092	0.823	no
1.2	0.261	0.902	0.330	0.092	0.823	no
1.5	**0.271**	0.869	0.363	0.092	0.823	yes
1.8	0.260	0.836	0.394	0.092	0.823	yes
2.2	0.254	0.793	0.429	0.092	0.823	yes
2.6	0.252	0.754	0.461	0.092	0.823	yes

Prior-blending coefficient. Table B.11 reports the sweep over the prior-blending coefficient α. The regime block is essentially insensitive to α for both commodities: nickel PR-AUC varies by only 0.001 across the valid range $\alpha \in \{0.10, 0.20, 0.30\}$ (0.269–0.270), and copper PR-AUC varies by 0.006 across the entire swept range $\alpha \in \{0.00, 0.05, 0.10, 0.20, 0.30\}$ (0.255–0.261). For nickel, the lowest two values ($\alpha = 0$ and $\alpha = 0.05$) are rejected by the entropy gate but yield essentially the same PR-AUC as the valid configurations, illustrating that the gate enforces

a structural constraint on regime representation quality rather than a tuning of predictive metrics. In aggregate, the prior-blending coefficient is the least consequential of the three knobs, and the baseline value of $\alpha = 0.10$ is the smallest blend that simultaneously satisfies all health constraints for nickel.

Table B.11: Marginal sweep over the prior-blending coefficient α, with the anchor year fixed at 2019 and $T = 1.8$. Columns as in Table B.9.

α	PR-AUC	Med. max prob	Norm. entropy	Min cluster share	Pre-2019 conc.	Valid
Nickel						
0.00	0.273	0.930	0.259	0.094	0.574	no
0.05	0.273	0.903	0.353	0.092	0.574	no
0.10	**0.270**	0.873	0.424	0.092	0.569	yes
0.20	**0.270**	0.808	0.537	0.084	0.565	yes
0.30	0.269	0.737	0.627	0.081	0.565	yes
Copper						
0.00	0.255	0.885	0.239	0.096	0.823	yes
0.05	0.260	0.861	0.327	0.092	0.823	yes
0.10	0.260	0.836	0.394	0.092	0.823	yes
0.20	0.256	0.786	0.504	0.092	0.823	yes
0.30	**0.261**	0.733	0.595	0.091	0.823	yes

Summary. The three swept hyperparameters exhibit distinct sensitivity profiles. The anchor year is the most consequential: it determines whether the resulting regime partition is structurally valid at all, and within the valid set it produces the largest spread in regime-only PR-AUC. The chosen 2019 anchor coincides with the empirical optimum among the valid options for nickel and is the only valid option for copper. This convergence between the structural-break argument for 2019 (Section 4.2, Chapter 4) and the data-driven sensitivity result provides independent support for the anchor choice. The posterior-softening temperature and the prior-blending coefficient are both highly stable within their respective valid ranges, with the baseline values of $T = 1.8$ and $\alpha = 0.10$ producing PR-AUC indistinguishable from neighbouring valid configurations (variation under 0.020). Configurations falling outside the valid region are uniformly characterised by entropy or concentration violations rather than poor predictive performance, reflecting the role of the health gates in excluding degenerate clustering solutions rather than tuning predictive metrics directly.

B.9 Regression-Layer Ablation Details

The main chapter reports the core regression fit and interval metrics. This appendix section retains only the additional ablation detail for the regression stage itself.

The two regressor ablations behave very differently. The disruption-focused sample weighting is clearly load-bearing: removing it degrades firing-week fit on every task, most sharply for nickel, where downside R^2 falls from 0.554 to 0.187 and upside from 0.463 to 0.143. By contrast, the classifier-probability feature is close to neutral for the regressor, leaving MAE and R^2 essentially unchanged in both metals.

Table B.12: Regressor-only ablations on the alert-firing population. Both variants leave classifier discrimination and alert utility identical to the baseline (cf. Table 5.2) and act solely on the magnitude estimates. One variant removes the regressor's disruption-focused sample weighting, and the other removes the classifier-probability feature supplied to the regressor. MAE is in percentage return units; positive Bias indicates under-prediction.

		Down (firing)			Up (firing)		
Metal	**Variant**	**MAE**	R^2	**Bias**	**MAE**	R^2	**Bias**
Copper	Baseline	0.069	−0.323	−0.060	0.048	0.045	−0.004
Copper	No sample weighting	0.071	−0.475	−0.065	0.053	−0.174	0.037
Copper	No classifier-prob. input	0.070	−0.272	−0.058	0.048	0.073	−0.002
Nickel	Baseline	0.042	0.554	−0.009	0.071	0.463	0.023
Nickel	No sample weighting	0.065	0.187	−0.045	0.084	0.143	0.065
Nickel	No classifier-prob. input	0.043	0.563	−0.010	0.073	0.404	0.031

B.10 Classifier Architecture Ablation

The production classifier uses a two-round cross-aware twin structure (Section 4.2). To test whether this added structure is warranted, we compare it against two simpler architectures on nickel, holding the walk-forward folds and the tuned alert policy fixed so that differences are attributable to the classifier alone. The two variants are chosen to isolate the two design decisions independently: the single-stage variant removes the second stage while keeping twin directional models, isolating the value of the cross-aware retraining; the joint model removes the twin structure while keeping a single training pass, isolating the value of modelling the two directions separately.

- **Reference**: the two-round cross-aware twin classifiers.
- **Single-stage**: Round 1 only: independent down/up classifiers on the base feature set, with neither the cross-aware features nor the Round 2 retrain.
- **Joint multiclass**: a single XGBoost booster with a `multi:softprob` objective over the three classes {none, down, up}, from which per-direction probabilities are recovered.

Table B.13 reports the results. On the primary reference alert-utility metric the reference design dominates, exceeding the single-stage variant by 12.2% and the joint model by 31.7%. The joint model attains marginally higher average PR-AUC across both directions, confirming it is a slightly sharper discriminator, but this does not convert into operational value: the staged twin structure extracts substantially more captured directional variation per alert, justifying the additional training complexity of the two-round design.

Table B.13: Nickel classifier architecture ablation. Alert utility is the total over both directions (basis points of captured directional variation, net of penalties); PR-AUC is reported per direction. Best per column in bold.

Architecture	**Alert utility**	**PR-AUC (down)**	**PR-AUC (up)**
Reference (two-round twin)	**200.3**	**0.522**	0.409
Single-stage (Round 1 only)	178.5	0.479	0.538
Joint multiclass	152.1	0.495	**0.551**

A plausible explanation for this divergence is that the joint softmax must distribute probability mass across three mutually exclusive classes, which can sharpen relative ranking (raising PR-AUC) while compressing the per-direction probabilities that the alert policy thresholds against, lowering recall at the operating point. The twin classifiers, by contrast, can each specialise on their own direction and fire independently, capturing more directional variation per alert. Under the stricter fixed-policy re-score of Appendix B.5, the same architecture snapshots score 133.5 for the reference twin model, 89.9 for the joint multiclass model, and 62.9 for the single-stage model; this preserves the reference model's lead while showing that the relative ordering of the two simplified alternatives is sensitive to the penalty convention.

B.11 SHAP Attribution Analysis

SHAP remains valuable here, but its role is interpretive rather than dispositive. SHAP tells us what the fitted baseline models use; the ablations tell us what actually adds marginal value out of sample. Those are related questions, but they are not the same question.

Table B.14: SHAP attribution by feature family for the copper disruption classifiers (OOF), as a percentage of total mean absolute SHAP contribution. Families are listed in descending order of combined attribution across both directions.

Feature family	**Down (%)**	**Up (%)**
Macro	64.6	68.8
GDELT sentiment	-	18.7
Cross-metal transfer	15.4	-
Regime	10.1	12.4
Price and technical	9.9	-

Table B.15: SHAP attribution by feature family for the nickel disruption classifiers (OOF), as a percentage of total mean absolute SHAP contribution. Families are listed in descending order of combined attribution across both directions.

Feature family	**Down (%)**	**Up (%)**
Macro	42.0	81.1
Regime	34.0	5.6
GDELT sentiment	8.8	2.7
Price and technical	6.6	8.1
Cross-metal transfer	7.6	2.5
Inventory and positioning	0.9	-

The SHAP summaries in Tables B.14 and B.15 still support a coherent qualitative reading of the models. For nickel, Cross-metal transfer denotes not only copper-side covariates but also transferred signal from the copper classifier, including OOF probabilities. Macro features dominate both metals, while the secondary importance shifts by direction: copper-up assigns more weight to GDELT-sensitive signals, copper-down leans more on cross-metal transfer, and nickel-down shows a much more material role for regime context than nickel-up.

At the same time, the ablation results force a more careful interpretation. A family can have visible SHAP weight and still fail the marginal-value test once retraining, threshold selection, and alert scoring are applied. For that reason, the SHAP evidence should be read as consistency

evidence: it helps explain why the model behaves as it does, but it does not by itself justify retaining a block in the final production stack.

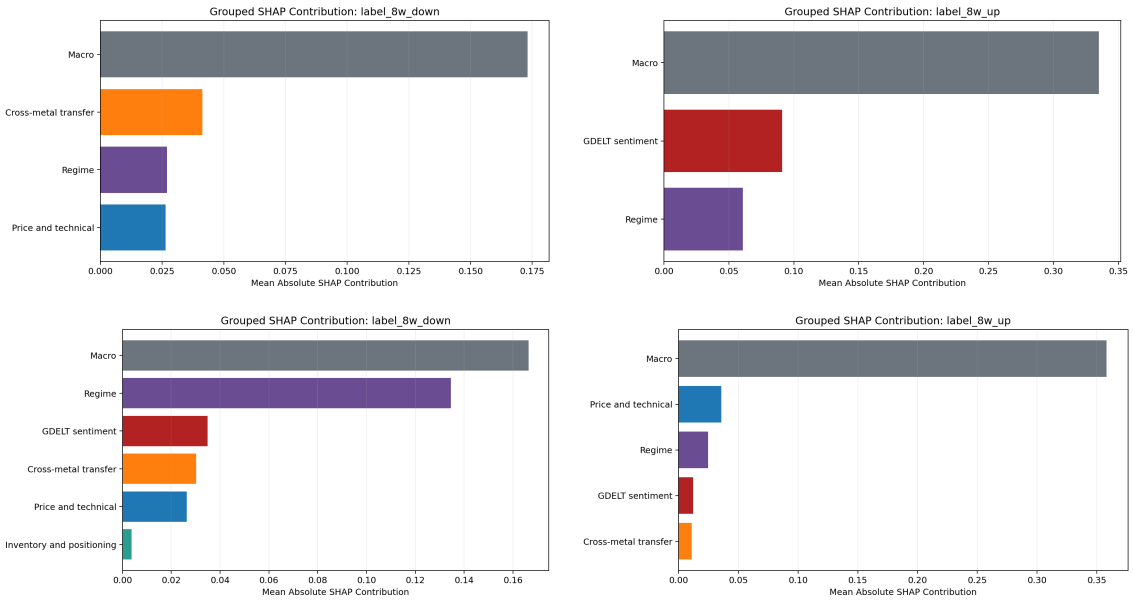

Figure B.5: Mean absolute SHAP contribution by feature family for the commodity classifiers (OOF), expressed as a percentage of each target's total. Top row: copper downside and copper upside. Bottom row: nickel downside and nickel upside.

Appendix C

Additional Manufacturing-Order Evaluation Diagnostics

C.1 First-Stage Classifier Diagnostics

The first-stage binary classifier distinguishes same-day orders from multi-day orders. Among the 3,249 test orders, 1,932 (59.5%) have a multi-day actual duration and 1,317 (40.5%) complete within a single working day. These are the realised populations; the classifier's routing decision is distinct from them. At the operating threshold the classifier routes 2,103 orders to the regressor and returns the remaining 1,146 immediately with a zero-day estimate without invoking the regressor. Routing is imperfect, so the two routed populations differ from the realised split by the classifier's errors: 238 genuinely same-day orders are still sent to the regressor (false positives) and 67 multi-day orders are returned as same-day (false negatives).

Table C.1 reports classifier performance on the test set. The model achieves a precision-recall area under the curve (PR-AUC) of 0.977. At the operating threshold of 0.284, selected as the mean optimal threshold across cross-validation folds, it attains an $F_{\beta=1.5}$ score of 0.940 and an F_1 score of 0.924 on the multi-day class.

Table C.1: First-stage classifier performance on the test set ($n = 3{,}249$). $F_{\beta=1.5}$ weights recall more heavily than precision, reflecting the asymmetric cost of misclassifying a multi-day order as same-day.

Metric	Value
Test-set size	3,249
Multi-day orders	1,932 (59.5%)
Decision threshold	0.284
PR-AUC	0.977
$F_{\beta=1.5}$ (multi-day class)	0.940
F_1 (multi-day class)	0.924
False negatives (multi-day $\rightarrow$ same-day)	67
Average missed duration	2.81 working days
False positives (same-day $\rightarrow$ regressor)	238

The classifier produces 67 false negatives, carrying a mean duration of 2.81 working days and implying a systematic underestimate of nearly three days for a small fraction of orders. The 238

false positives are less consequential operationally, as the regressor is expected to return a low estimate for orders whose features indicate minimal work content. The precision-recall curve is shown in Figure C.1; the steep initial rise indicates that high precision is achievable at moderate recall levels.

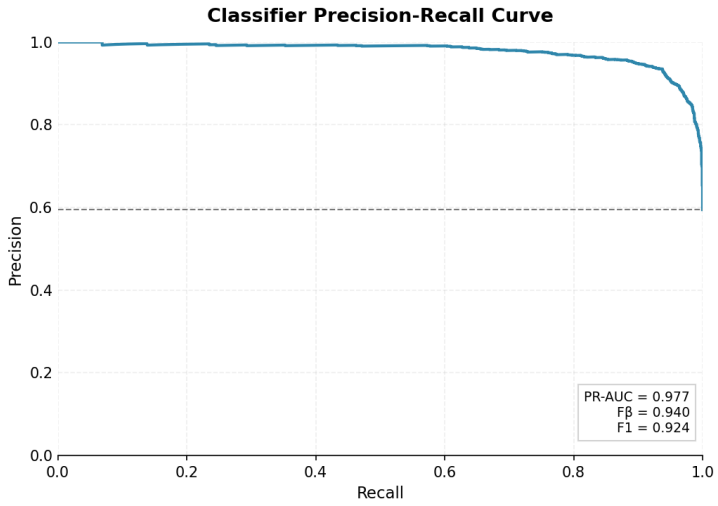

Figure C.1: Precision-recall curve for the first-stage classifier on the test set. The dashed horizontal line marks the prevalence of multi-day orders (0.595). The operating point at threshold 0.284 is marked. PR-AUC = 0.977.

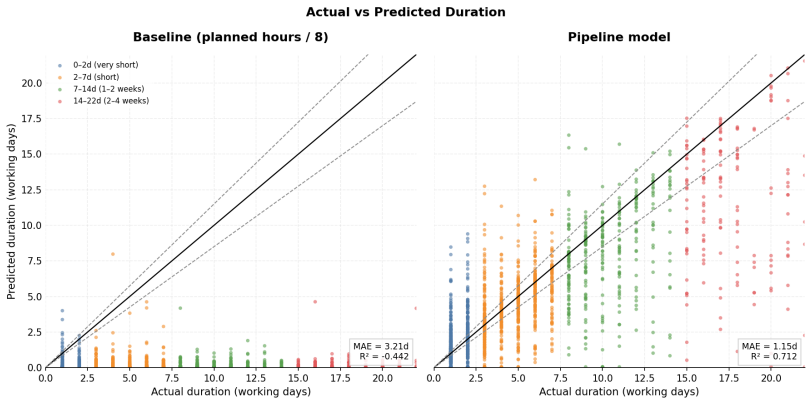

Figure C.2: Predicted versus actual duration for the full pipeline on the test set ($n = 3{,}249$). Points on the diagonal indicate perfect prediction. The dashed lines bound the $\pm 15\%$ within-band region. Colour intensity indicates point density.

C.2 Residual and Subgroup Diagnostics

Table C.2 reports pipeline performance by workstation category for all test orders, including those routed to the same-day estimate.

Results reveal pronounced heterogeneity across workstation types. The best-performing categories by rMAE% are the project mounting workstation (12.2%), the sawing workstation (19.5%),

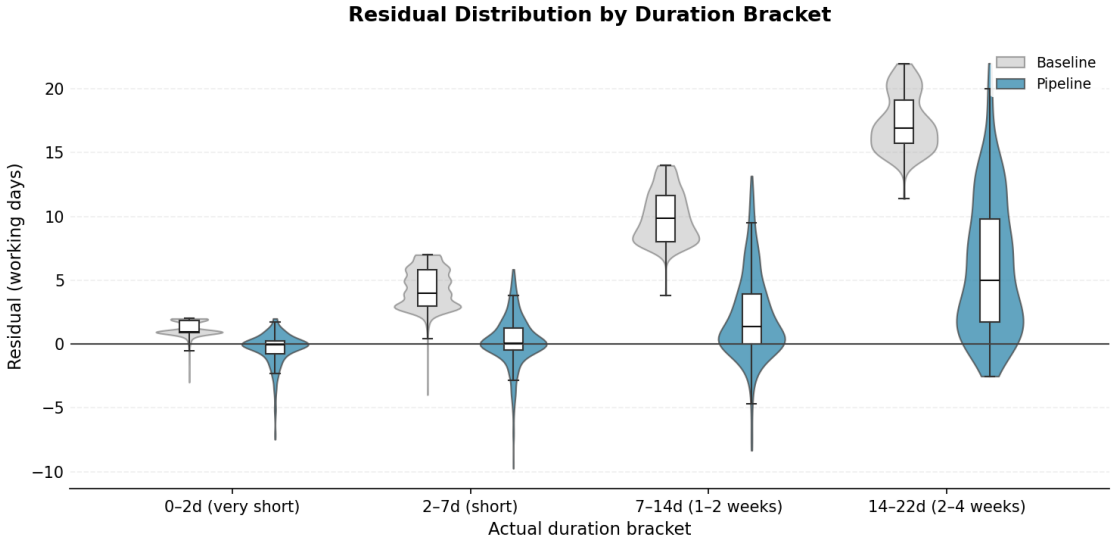

Figure C.3: Distribution of signed residuals (actual − predicted, working days) for the planned-hours-derived estimate (left, gray fill) and pipeline (right, coloured fill) within each duration bracket. The horizontal line at zero marks unbiased prediction. Bracket sample sizes are as reported in Table 5.6.

Table C.2: Pipeline performance by primary workstation type on the test set. For orders routed through multiple workstation types (*Mixed*), the multi-type routing itself is treated as the workstation segment. rMAE% is normalised by the training-group median for each category. The Automation segment ($n = 13$) is too small for reliable inference and is included for completeness only.

Workstation type	n	MAE (d)	rMAE%	R^2	Bias (d)	InBand (%)
Mixed (multi-type)	1,766	1.651	41.3	0.673	0.504	41.9
Laser	311	1.027	102.7	0.467	0.589	48.2
Project Mounting	261	0.122	12.2	−0.214	0.036	96.9
Welding	199	0.582	58.2	0.092	0.457	76.9
Sawing	195	0.195	19.5	0.177	0.144	87.7
Machining	173	0.999	99.9	0.761	0.194	49.1
Other	166	0.316	31.6	0.527	0.195	80.7
Assembly	115	0.482	48.2	0.251	0.247	75.7
Bending	50	0.353	35.3	0.267	0.260	90.0
Automation	13	0.702	70.2	−12.049	−0.548	76.9
Overall	3,249	1.148	47.5	0.712	0.401	56.3

and the bending workstation (35.3%), all achieving within-band accuracies above 80%. The project mounting workstation is particularly well-characterised: its consistent process structure means that duration correlates strongly with allocated component volume and planned work content, yielding an MAE of only 0.122 working days. Laser cutting (rMAE% = 102.7%) and welding (rMAE% = 58.2%) show higher relative errors, attributable to operator-level and material-level variability not fully represented in the available features. The large multi-type routing segment (rMAE% = 41.3%, $R^2 = 0.673$) is broadly consistent with overall pipeline performance, with routing-composition features partially capturing the greater complexity of multi-workstation orders. Figure C.4 provides a visual comparison of pipeline versus the planned-hours-derived estimate rMAE% by workstation type.

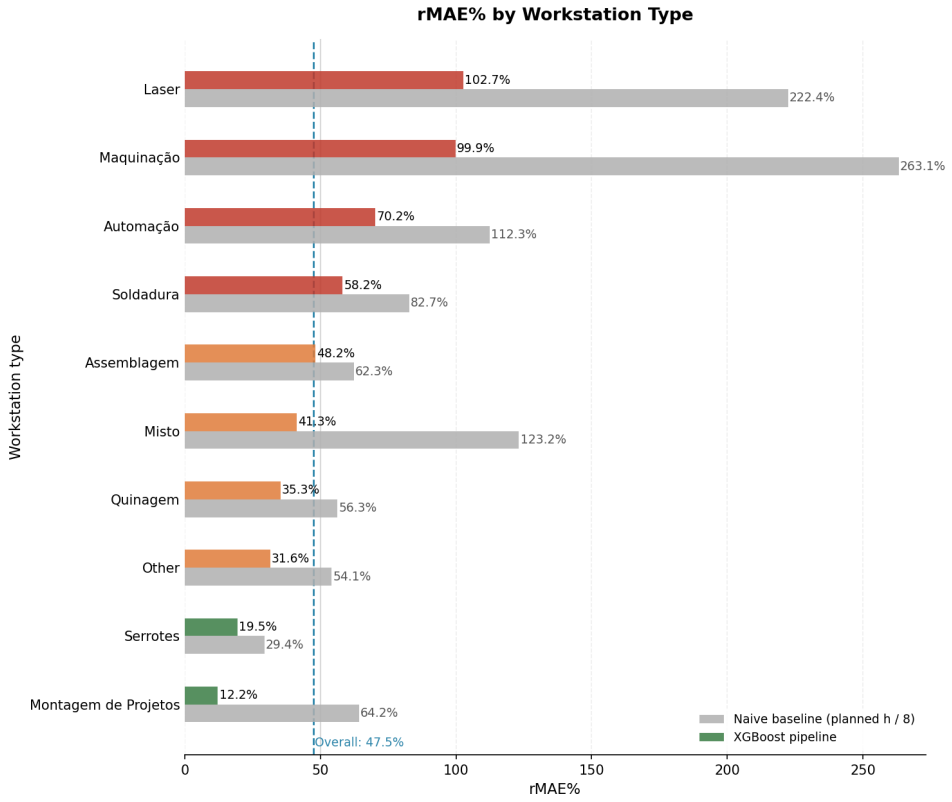

Figure C.4: Relative MAE (%) by workstation type for the pipeline (coloured bars) and the planned-hours-derived estimate (gray bars). The vertical dashed line marks the overall pipeline rMAE%.

Table C.3 reports pipeline performance by project type.

External client orders, comprising 76.9% of the test set, are predicted with the highest accuracy ($R^2 = 0.748$, rMAE% = 44.3%), consistent with the denser historical population available for this category and the greater informativeness of part-level history features when prior order volume is high.

Stock replenishment orders exhibit the highest MAE (2.737 days) and the lowest R^2 (0.300). Because these orders are launched to fill buffer inventory rather than in direct response to a customer requirement, they are more susceptible to opportunistic re-sequencing; the positive bias of 1.012 days is consistent with queue-related delays that exceed what routing content alone predicts. Kanban-driven orders show similarly elevated rMAE% (72.6%) and reduced within-band

Table C.3: Pipeline performance by project type on the test set. Project types are ordered by descending sample size.

Project type	n	MAE (d)	rMAE%	R^2	Bias (d)	InBand (%)
External client orders	2,497	1.089	44.3	0.748	0.360	57.9
Direct sales orders	513	0.958	44.5	0.652	0.400	58.3
Stock replenishment	115	2.737	109.3	0.300	1.012	29.6
Kanban-driven orders	105	1.835	72.6	0.482	0.751	37.1
Internal project orders	19	0.619	26.0	0.753	0.136	57.9
Overall	3,249	1.148	47.5	0.712	0.401	56.3

accuracy (37.1%) for related reasons: their scheduling is governed by replenishment triggers rather than hard deadlines, making them more susceptible to deprioritisation during periods of shop-floor congestion. Figure C.5 illustrates the rMAE% comparison by project type.

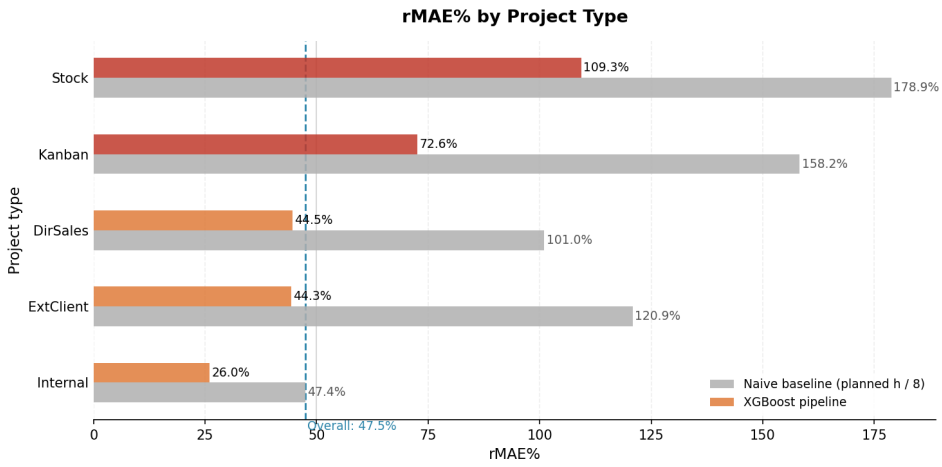

Figure C.5: Relative MAE (%) by project type for the pipeline (coloured bars) and the planned-hours-derived estimate (gray bars). The overall pipeline rMAE% is indicated by the dashed line.

C.3 Hyperparameter Search and Selection

The classifier, regressor, and residual-quantile models are each tuned by grid search under the five-fold cross-validation scheme of Section 4.3. The same candidate grid is searched for all three models, and each is scored against its own selection criterion: the F_β score ($\beta = 1.5$) on the non-trivial class for the classifier, day-space MAE after lognormal bias correction for the regressor, and the mean pinball loss across the three quantile levels for the residual-quantile model. The number of boosting rounds is not part of the grid; it is determined per candidate by early stopping on the validation fold with a patience of 50 rounds. The L1 and L2 regularisation strengths were fixed at previously tuned per-model values rather than re-searched, keeping the grid tractable given the depth of the trees involved. Table C.4 reports the search grid and the values selected for each model.

Table C.4: Hyperparameter search grid and selected values for the manufacturing-order duration models. The same grid is searched for all three models by five-fold cross-validation against the per-model criterion below; the number of boosting rounds is set by early stopping (patience 50, ceiling 20,000) rather than grid-searched.

		Selected		
Hyperparameter	**Search grid**	**Classifier**	**Regressor**	**Quantile**
Maximum depth	{9, 13, 17, 21, 25}	25	17	21
Learning rate	{0.03, 0.05}	0.05	0.03	0.03
Min. child weight	{5, 10, 15}	5	5	15
Column subsample	{0.8, 1.0}	1.0	1.0	0.8

Selection criteria: classifier, F_β with $\beta = 1.5$ on the non-trivial class; regressor, day-space MAE after lognormal bias correction; residual-quantile, mean pinball loss across the three levels ($\alpha \in \{0.1, 0.5, 0.9\}$). *Fixed for all candidates:* row subsample 0.8; Regularisers held at previously tuned per-model values (classifier λ=1.0, α=0.0; regressor and residual-quantile λ=5.0, α=1.0); objectives logistic loss (classifier), absolute-error loss (regressor), and quantile (pinball) loss (residual-quantile model); the regressor's base score is set to the training-set median log-duration.

C.4 Feature-Group Ablation

The SHAP attribution reported in Section 5.2 measures how the *fitted* model distributes credit across feature groups. Attribution is not, however, the same as marginal predictive value: a group may receive high attribution yet be redundant once the model is retrained without it, or carry modest attribution yet prove load-bearing. To measure marginal value directly, each of the six feature groups defined in Section 4.3 is removed in turn and the full two-stage pipeline is retrained from scratch on the training split with that group excluded *before* the SHAP pruning step, so the model is free to compensate with the remaining features. Each retrained variant is then scored on the same held-out test set using the metrics of Table 5.5, and reported relative to the all-features baseline.

Hyperparameters are held fixed at the values selected for the baseline model (Section 4.3); only the number of boosting rounds is re-determined per variant through early stopping. This keeps each variant's ΔMAE attributable to the removed features rather than to re-optimised model capacity. It also differs deliberately from the commodity ablation (Appendix B), which re-runs its hyperparameter search for every variant: each ablation reproduces the training procedure of its own model.

Because the six groups differ substantially in size, a leave-one-group-out design confounds a group's importance with its feature count. Removing a 35-feature group withdraws more information than removing a 15-feature group irrespective of per-feature value. The table therefore reports both the raw ΔMAE and the size-normalised effect per feature (ΔMAE/ft), which should be read together. A positive ΔMAE (equivalently a negative ΔR^2) indicates that removing the group *degrades* accuracy, i.e. the group earns its place in the model.

Removing the historical duration signals group produces the largest degradation (ΔMAE $=$ $+0.112$ working days, $\Delta R^2 = -0.048$), followed by component availability ($+0.085$) and shop-floor load at launch ($+0.069$). The size-normalised view tells a complementary story: the compact shop-floor-load group contributes the most *per feature* (ΔMAE/ft $= +0.0046$ across only 15 features), essentially tied with historical duration signals ($+0.0045$), so its more modest raw effect reflects its small size rather than weak signal. At the other end, scope of work is essentially redundant (ΔMAE $= -0.001$): its planned-work and complexity features are recoverable from the remaining groups on retraining. Most strikingly, workstation characterisation carries the largest

Table C.5: Manufacturing-order duration feature-group ablation on the held-out test set. Each row removes one feature group and retrains the full two-stage pipeline. #grp is the number of features in the respective group; ΔMAE and ΔR^2 are relative to the all-features baseline; ΔMAE/ft normalises the effect by group size. Positive ΔMAE / negative ΔR^2 means the group earns its place.

Removed group	**#grp**	**MAE**	**ΔMAE**	**ΔMAE/ft**	R^2	ΔR^2	**rMAE%**
Baseline (all features)	151	1.148			0.712		47.5
Historical Duration Signals	25	1.260	+0.112	+0.0045	0.664	−0.048	50.9
Component Availability	33	1.232	+0.085	+0.0026	0.687	−0.026	51.9
Shop-Floor Load at Launch	15	1.216	+0.069	+0.0046	0.692	−0.020	50.1
Order Context & Position	26	1.162	+0.014	+0.0006	0.710	−0.003	48.2
Workstation Characterisation	35	1.156	+0.008	+0.0002	0.708	−0.004	47.8
Scope of Work	17	1.147	−0.001	−0.0001	0.715	+0.003	47.8

SHAP share (40.4%, Figure 5.1) yet contributes only +0.008 days to MAE under ablation; the ordering by marginal contribution therefore differs sharply from the SHAP attribution ordering. This reinforces that attribution and marginal predictive value are distinct, and that feature retention is best judged by the latter.

C.5 Duration Prediction Interval Diagnostics

In addition to point-prediction accuracy, the manufacturing-order duration pipeline was evaluated in terms of interval calibration. For non-trivial orders routed to the regressor, the residual-quantile model produces P10 and P90 predictive bounds around the central estimate. Under perfect calibration, this interval would be expected to contain the realised duration in approximately 80% of cases. The interval diagnostics are evaluated on the regressor-routed subset only, and exclude regressor-routed cases whose realised duration was exactly zero working days.

These observations remain relevant when assessing the classifier, since they represent routing false positives, but they are not informative about the intended behaviour of the uncertainty layer: a same-day completion is, by construction, outside the use case for a positive-duration regressor interval.

After this filtering, the empirical P10–P90 coverage on the test set is 63.4% over 1,865 regressor-routed orders. The remaining cases are split between 12.5% of realised durations falling below the lower bound and 24.0% falling above the upper bound. Mean interval width is 3.21 working days, with a median width of 2.63 working days. The residual-quantile construction produces materially tighter intervals while preserving similar overall calibration, but still undercovers relative to the nominal 80% target.

The behaviour of the intervals remains heterogeneous across the duration range. In the 1–2 day bracket, coverage is 65.7% on a narrow mean width of 1.84 working days, with the misses split roughly evenly between 16.9% of realised durations below P10 and 17.4% above P90. Coverage is 64.0% in the 3–7 day segment and 67.6% for 8–14 day orders, while it falls to 43.0% for 15–22 day orders. In this longest bracket the misses are overwhelmingly realised durations falling above the upper bound (55.1%), which is consistent with the positive-bias pattern reported in the main results and indicates that the model still underestimates the longest and most operationally variable jobs even after the interval adjustments.

Temporal diagnostics suggest that interval quality is stable across the test horizon. For orders starting between 2025-06-02 and 2025-10-13, empirical coverage is 64.1% with a mean interval

width of 3.62 days. For orders starting between 2025-10-13 and 2026-03-13, coverage is 62.8% with a mean interval width of 2.80 days. This suggests no major temporal collapse in uncertainty quality over the evaluation window, although the intervals remain underconservative overall.

Table C.6: Overall P10–P90 interval diagnostics for regressor-routed manufacturing orders, excluding regressor-routed cases with realised zero-day duration.

Subset	n	Coverage (%)	Below P10 (%)	Above P90 (%)	Mean width (days)
Overall	1,865	63.4	12.5	24.0	3.21

Table C.7: P10–P90 interval diagnostics by realised duration bracket, excluding regressor-routed cases with realised zero-day duration.

Duration bracket	n	Coverage (%)	Below P10 (%)	Above P90 (%)	Mean width (days)
1–2 days	673	65.7	16.9	17.4	1.84
3–7 days	713	64.0	14.0	22.0	3.28
8–14 days	321	67.6	5.3	27.1	4.65
15–22 days	158	43.0	1.9	55.1	5.79

Table C.8: Temporal stability of duration-interval diagnostics after excluding regressor-routed cases with realised zero-day duration.

Period	n	Coverage (%)	Below P10 (%)	Above P90 (%)	Mean width (days)
2025-06-02 to 2025-10-13	932	64.1	13.0	23.0	3.62
2025-10-13 to 2026-03-13	933	62.8	12.1	25.1	2.80

C.6 Comparison of Prediction-Interval Methods

The residual-quantile construction adopted in the deployed pipeline is one of several ways to attach an 80% P10–P90 interval to the point prediction. This section compares it against three alternatives on the same hold-out evaluation population (the 1,865 regressor-routed orders with non-zero realised duration used above), so that the methods differ only in how the interval is formed and not in the data they are scored on. The methods considered are:

- **Isotonic** : a separate quantile model whose P10, P50 and P90 outputs are each recalibrated by an isotonic regression fitted on out-of-fold predictions.
- **Split-Conformal** : the same quantile model with asymmetric split-conformal tail adjustments calibrated to the nominal level on out-of-fold nonconformity scores.
- **CQR** : conformalized quantile regression (Romano et al., 2019), which applies a single symmetric conformal correction $\hat{q}$ to both quantile boundaries.
- **Residual-Quantile** : the deployed method: a quantile model (Koenker & Bassett, 1978) trained on the residual actual $-$ point, reconstructed as point $+$ residual quantile.

Table C.9: Overall interval-method comparison on the regressor-routed test population ($n = 1{,}865$). Coverage targets 80%. <P10 and >P90 are the fractions of realised durations below the lower and above the upper bound; W. mean and W. med. are the mean and median interval width in working days. Lower pinball loss is better.

Method	Cov. (%)	<P10 (%)	>P90 (%)	W. mean	W. med.	Pinball
Isotonic	44.1	30.5	25.4	2.64	2.21	0.787
Split-Conformal	75.7	9.8	14.6	5.13	4.36	0.633
CQR	73.4	7.2	19.4	4.77	3.99	0.645
Residual-Quantile	63.4	12.5	24.0	3.21	2.63	**0.632**

The comparison is more nuanced than the headline coverage figures suggest, and it motivates the choice of the residual-quantile method despite its lower marginal coverage. Two observations stand out.

First, marginal coverage alone is a misleading basis for selection. The two conformal methods come closest to the nominal 80% marginally (Split-Conformal 75.7%, CQR 73.4%) but still fall short of it, and they reach even that level only by inflating the intervals to a mean width of roughly five working days, some 50–60% wider than the residual-quantile band (3.21 days). Their additional coverage is purchased with intervals that are too wide to be operationally informative for the short orders that dominate the factory mix.

Second, and more importantly, the marginal guarantee hides strongly uneven *conditional* behaviour (Table C.10). Both conformal methods over-cover the short and medium brackets (Split-Conformal and CQR reach 78–84% on orders up to seven days) while collapsing on the long tail, where CQR covers only 32.9% of 15–22 day orders and Split-Conformal 41.8%. In other words, they spread their coverage by being far too wide on the easy majority while still failing the hard cases. This is expected: split-conformal and CQR guarantee *marginal* coverage under exchangeability and apply an essentially constant width adjustment, which cannot adapt to the sharp growth in duration variance across brackets.

The residual-quantile method behaves differently. Its marginal coverage is lower (63.4%), but it attains the best pinball loss of all methods (0.632, narrowly ahead of split-conformal's 0.633 and against 0.65–0.79 for the others), the tightest informative intervals among the calibrated methods, and the most uniform conditional coverage (65.7/64.0/67.6/43.0 across the four brackets). Because it conditions the residual quantiles on the point prediction and the classifier margin, it adapts its width to the order rather than applying a global correction. The isotonic baseline, by contrast, is dominated throughout, undercovering everywhere (44.1% marginally).

The one regime that defeats every method is the 15–22 day tail, where no method exceeds 43% conditional coverage. This reinforces the point made in the main results: the drivers of the

Table C.10: Conditional P10–P90 coverage (%) by realised duration bracket for each interval method, on the same population as Table C.9. The nominal target is 80% within every bracket.

Method	1–2 days	3–7 days	8–14 days	15–22 days
Isotonic	42.4	48.4	47.7	24.7
Split-Conformal	77.7	83.6	70.4	41.8
CQR	82.5	78.3	63.2	32.9
Residual-Quantile	65.7	64.0	67.6	43.0

longest orders are systematically under-observed in the ERP feature set, so the limitation there is one of information rather than of interval construction. Taken together, the comparison supports selecting the residual-quantile method on the basis of sharpness and conditional consistency, and treating marginal coverage as a necessary but insufficient diagnostic.

C.7 Temporal Stability

To assess whether predictive performance degrades over time, the test set is partitioned into two chronological halves of equal size and evaluated independently. Table C.11 reports the results.

Table C.11: Pipeline performance on the first and second chronological halves of the test set. The split date falls on 14 October 2025.

Period	n	Date range	MAE (d)	rMAE%	R^2	Bias (d)	InBand (%)
First half	1,624	2 Jun – 13 Oct 2025	1.280	51.0	0.725	0.455	55.2
Second half	1,625	14 Oct 2025 – 16 Mar 2026	1.015	43.9	0.687	0.346	57.3

Performance is broadly consistent across both halves, with MAE improving from 1.280 to 1.015 days and both R^2 values close to the overall figure of 0.712. The lower bias in the second half (0.346 vs. 0.455 days) is consistent with the expanding-window feature construction, in which the most recent historical signals become progressively more representative of current factory conditions. These results indicate no systematic performance degradation over the ten-month test window.

Appendix D

Shop-Floor Simulation: Implementation Details

D.1 Operation-Time Representation

Each manufacturing order is represented by a dynamic state record tracking its current operation index, the workstation type of the current operation, the hourly demand that operation places on the workstation per simulated day, and the fractional number of days of work remaining before the current operation completes.

Let h_o denote the corrected planned hours for operation o and let $H = 8$ be the reference working hours per day. The number of working days allocated to an operation is:

$$d_{\text{alloc}}(o) = \max\left(\varepsilon,\ \left\lceil \frac{h_o}{H} \right\rceil\right) \tag{D.1}$$

where $\varepsilon > 0$ is a small positive floor that handles zero-hour operations so that the simulator never assigns a zero-length slot. The daily demand that operation o imposes on its workstation is then:

$$\delta(o) = \frac{h_o}{d_{\text{alloc}}(o)} \tag{D.2}$$

This formulation allocates a minimum of one full simulated day to any operation whose planned hours are less than H. A two-hour operation therefore occupies one simulated day, while a twenty-hour operation occupies $\lceil 20/8 \rceil = 3$ days. Because daily demand is proportional to planned hours, a two-hour operation contributes only $2/8 = 0.25$ units of hourly load per day, leaving the remaining capacity available to other orders.

D.2 Daily Advancement Mechanics

At each daily tick, orders advance through their routing via a fractional-day consumption loop allocated a budget of one full simulated day. At each iteration, let r denote the remaining allocated days on the current operation and b the remaining day budget. The time consumed is:

$$c = \min(r,\ b) \tag{D.3}$$

with state updates:

$$r \leftarrow r - c, \qquad b \leftarrow b - c \tag{D.4}$$

When $r \leq 0$, the operation completes. The workstation load table is updated to remove the previous demand and add the next operation's contribution. Formally, when transitioning from operation o_i to o_{i+1}:

$$L_{w_i} \leftarrow L_{w_i} - \delta(o_i), \qquad L_{w_{i+1}} \leftarrow L_{w_{i+1}} + \delta(o_{i+1}) \tag{D.5}$$

where L_w denotes the current load on workstation w. The loop continues with the residual budget on the new operation, allowing a single order to complete multiple short operations within one simulated day. When the final operation is exhausted, the order exits the active set and its produced material is credited to the stock register.

D.3 Synthetic Arrivals and Material Eligibility

A simulation horizon that adapts to the size and complexity of the candidate set is long enough that new manufacturing orders will enter the shop floor during the period being modelled. Without accounting for this, workstation loads would decline monotonically as existing orders complete and no new demand enters, producing an unrealistically optimistic picture of future capacity. To correct for this, the simulation injects synthetic orders on each day of the horizon according to empirically derived arrival profiles.

An arrival profile is constructed for each distinct workstation by querying the historical record of completed operations over a recent lookback window. For each working day d in that window and each workstation type w, the query records the count of manufacturing orders whose first operation was at w on day d, the total planned hours carried by those orders across their full routing, and the average number of working days elapsed between their launch date and their planned first operation start. Denoting the resulting daily series of order counts as $\{n_w(d)\}_{d \in \mathscr{D}_w}$, the daily expected arrival count for workstation w is:

$$\lambda_w = \mathrm{P50}(\{n_w(d)\}_{d \in \mathscr{D}_w}) \tag{D.6}$$

Alongside the count, the profile stores the median total route hours $\bar{h}_w$ and a representative route template, defined as the most common workstation sequence with median planned hours per position. On each simulated day t, $\lfloor \lambda_w \rceil$ synthetic orders are injected for every workstation type with a non-zero profile; each order uses the template with hours scaled to $\bar{h}_w$ and a planned start offset by the median slack $\bar{\sigma}_w$:

$$t_{\text{plan}} = t + \bar{\sigma}_w \qquad \text{(working days)} \tag{D.7}$$

Injected orders enter the open background queue and compete with real orders and candidates for workstation capacity, ensuring that the simulated load trajectory reflects the expected rate of new work entering the factory.

Before a candidate may be scheduled to start on a given day, a stock eligibility check is applied. Let $\mathscr{R}(f)$ denote the set of required material references for candidate f, with required quantity $q_{f,m}$ for reference m, and let s_m denote the current simulated stock level. A candidate is considered theoretically eligible if and only if:

$$\text{eligible}(f) = \bigwedge_{m \in \mathscr{R}(f)} (s_m \geq q_{f,m}) \tag{D.8}$$

If any required material is unavailable in the required quantity, the order is flagged as stock-ineligible and assigned a severe scoring penalty that prevents it from being scheduled on that day. The stock register evolves during the simulation as background orders finish and as anticipated procurement arrivals from confirmed purchase orders are injected on their scheduled delivery dates.

D.4 Capacity Banding and Candidate Scoring

Throughput ratio derivation For each workstation type w and completed working day d, let $Q_w^{\text{start}}(d)$ and $Q_w^{\text{end}}(d)$ denote the queue backlog hours at the start and end of day d respectively. The daily throughput ratio is:

$$r_w(d) = \frac{Q_w^{\text{start}}(d) - Q_w^{\text{end}}(d)}{Q_w^{\text{start}}(d)} \tag{D.9}$$

The P50 capacity target ratio and the effective daily capacity target for workstation w on day t are:

$$\tau_w = \text{P50}(\{r_w(d)\}_{d \in \mathscr{D}_w}), \qquad C_w(t) = B_w(t) \cdot \tau_w \tag{D.10}$$

where $B_w(t)$ is the queue backlog at the start of day t and $\mathscr{D}_w$ is the set of recent working days in the lookback window.

Scoring penalty definitions The pressure penalty and gate penalties referenced in the composite score of Section 4.4 are defined as follows. The pressure penalty is:

$$\text{pen}(w,t) = \max\left(0,\ \frac{L_w(t)}{C_w(t)} - 1\right) \times \kappa \tag{D.11}$$

where $L_w(t)$ is the current load on workstation w at time t and $\kappa = 10$ is a fixed scaling constant. The gate penalties are:

$$G_{\text{stock}}(f,t) = \begin{cases} -100 & \text{if } \neg\text{eligible}(f,t) \\ 0 & \text{otherwise} \end{cases} \tag{D.12}$$

$$G_{\text{cap}}(f,t) = \begin{cases} -20 & \text{if capacity-blocked and fill-to-surpass does not apply} \\ 0 & \text{otherwise} \end{cases} \tag{D.13}$$

Candidate selection proceeds greedily in urgency-rank order. If a candidate passes both the stock eligibility and capacity gates, it is marked as scheduled and its demand is committed to the workstation load. If it fails either gate, it remains in the waiting pool and is re-evaluated on the next simulated day.

Start-date scoring and lineage planning The simulation produces, for each candidate, the earliest day on which it was successfully scheduled. This day defines the feasibility floor for the start-date recommendation. Let t^* denote this earliest feasible start day and let k index the number of additional working days of delay, so that the candidate start day is $t^* + k$. The cost of starting on day $t^* + k$ is:

$$\text{cost}(k) = \lambda_u(f) \cdot k + \alpha \cdot \rho_w(t^*{+}k) + \gamma \cdot n(t^*{+}k) + \beta \cdot \max\Big(0,\ \rho_w(t^*{+}k) - 0.85\Big) \cdot \hat{d}(f) \tag{D.14}$$

where $\rho_w(t) = L_w(t)/C_w(t)$ is the bottleneck load ratio at day t; $n(t)$ is the number of manufacturing orders concurrently active on the shop floor at day t; $\hat{d}(f)$ is the predicted duration of order f in working days; and $\alpha = 8.0$, $\gamma = 0.15$, and $\beta = 0.35$ are fixed weight constants for the bottleneck load, concurrent queue congestion, and duration-risk components respectively. The delay weight $\lambda_u(f)$ is urgency-adaptive:

$$\lambda_u(f) = \begin{cases} 2.4 & \text{if } s_b(f) \le 1, \\ 1.5 & \text{if } 2 \le s_b(f) \le 4, \\ 0.8 & \text{if } s_b(f) > 4, \end{cases}$$

where the slack buffer is $s_b(f) = \max\big(0,\ s(f) - \lceil \hat{d}(f) \rceil\big)$ and $s(f)$ denotes the number of working days between the planning date and the order's first planned operation start. The recommended start date corresponds to:

$$k^* = \underset{k \in \{0,1,2,3\}}{\arg\min}\ \text{cost}(k) \tag{D.15}$$

The forward window is fixed at three working days because delaying a start beyond that point to avoid congestion would itself constitute a meaningful scheduling cost.

For finish-date reporting, let $\theta = 0.30$ denote a fractional-day completion threshold and let the effective working-day span be

$$\tilde{d}(f) = \max\big(0,\ \lceil \hat{d}(f) - \theta \rceil\big).$$

This rounding absorbs sub-threshold fractional days into the current working day rather than spilling them into the next. The predicted finish date is:

$$t_{\text{finish}}(f) = t_{\text{start}}(f) + \tilde{d}(f) - 1 \qquad \text{(working days)} \tag{D.16}$$

When a manufacturing order has open child orders that must be completed before the parent order can begin, the parent's recommended start date is derived as the maximum of two lower bounds: the *dependency floor*, defined as the first working day following the latest predicted finish date among all open child orders, and the *procurement floor* $t_{\text{proc}}(\text{parent})$, defined as the earliest date on which all direct material requirements of the parent order itself are confirmed to be available:

$$t_{\text{start}}(\text{parent}) = \max\left(\max_{f \in C}\ t_{\text{finish}}(f) + 1,\ \ t_{\text{proc}}(\text{parent}) \right) \tag{D.17}$$

where C is the set of open child orders.

D.5 Throughput Dynamics Validation

The forward simulator was validated with a point-in-time backtest designed to assess whether the simulated shop-floor state evolves at a rate comparable to the realised factory. Twenty-four historical anchor dates were used. For each anchor, the open and ongoing manufacturing-order population was reconstructed using only information available at the anchor date, the simulator was advanced over 20-, 40-, and 60-working-day horizons, and the resulting trajectory was compared with the realised operation history.

The main validation quantity is cumulative factory drain: the sum of all queue hours drained by the simulator over the full horizon compared with the corresponding realised sum. This is the most relevant indicator for the planning use case, because it measures whether the simulation accompanies the real factory's aggregate productive movement over the horizon, rather than whether it assigns every hour to the same workstation and day as the historical record. Table D.1 reports this follow-through metric. Across all horizons, cumulative drained throughput has a relative mean absolute error (rMAE) of 22%, with a positive bias of 1,213.9 hours. The 20-working-day horizon, which is closest to the intended short-term planning use case, obtains an rMAE of 26% and a bias of 839.3 hours, indicating that the simulator overestimates total factory movement but preserves a meaningful aggregate relationship with the realised trajectory. The correlations remain positive, although they weaken as the horizon grows, so the relative error and bias are more stable indicators of horizon-level calibration.

Table D.1: Factory-level cumulative drained-throughput validation. Workstations are summed first, and each row compares the simulated and realised total drain over the complete anchor–horizon trajectory.

Horizon	MAE (h)	rMAE	Bias (h)	Corr.
All horizons	1,410.5	22%	1,213.9	0.39
20 wd	882.6	26%	839.3	0.53
40 wd	1,369.6	21%	1,274.2	0.40
60 wd	1,979.4	20%	1,528.3	0.25

Daily drained throughput is a stricter and noisier diagnostic. In Table D.2, the same workstation totals are compared day by day instead of cumulatively over the horizon. The rMAE rises to 62–63% and the correlation is close to zero. This does not contradict the stronger cumulative result: it indicates that the simulator captures the approximate amount of factory movement over a planning window, but not the exact historical day on which that movement occurred. Such daily discrepancies are expected in an engineer-to-order environment, where operator availability, interruptions, informal resequencing, delayed system registration, and partial completions introduce high day-level variability that is not fully observable in the planning data.

Table D.2: Factory-day validation after summing all workstations. This table measures day-by-day timing accuracy rather than cumulative horizon follow-through.

Horizon	MAE (h/day)	rMAE	Bias (h/day)	Corr.
All horizons	104.4	63%	33.1	0.03
20 wd	105.9	63%	42.0	0.04
40 wd	103.9	63%	31.9	0.03
60 wd	103.3	63%	25.5	0.01

Backlog accuracy was evaluated as a state variable rather than a flow variable. The initial factory backlog is recovered closely: the start-of-horizon rMAE is 7% and the correlation across anchors is 0.99.

The 7% start-of-horizon gap is definitional, not a seeding error: the realised series discounts in-progress operations using their *realised* completion dates, while the causally seeded simulator cannot, so it carries marginally more remaining work on operations already running at t_0. The end-of-horizon backlog error increases with the horizon, from 6% at 20 working days to 8% at 60 working days (Table D.3).

The positive bias means that the simulated state tends to retain more queue backlog than the realised system by the end of the simulation, even while cumulative drained throughput is also overestimated. The backlog state therefore reflects both throughput timing and the replenishment of the simulated floor, rather than a simple residual of total drain. The drift remains limited at the 20-working-day horizon, but grows and becomes less correlated with the realised final state as the simulation is extended further away from the point-in-time state reconstruction.

The workstation-disaggregated metrics are less favourable and are therefore interpreted as a secondary diagnostic rather than the primary validation criterion. Averaged over all workstations, days, and horizons, drained hours have a per-workstation-day rMAE of 63%, with a positive bias of 33.1 hours and a correlation of 0.03. This indicates that the simulator is not intended to reproduce the exact historical workstation-day allocation of each production hour. Instead, it is more

Table D.3: Factory-level backlog state accuracy at the start and end of each simulated horizon. Workstations are summed before computing the metrics.

State	Horizon	MAE (h)	rMAE	Bias (h)	Corr.
Start	All horizons	2,762.8	7%	2,762.8	0.99
End	20 wd	2,292.7	6%	1,535.1	0.85
End	40 wd	3,001.3	7%	1,658.3	0.58
End	60 wd	3,433.2	8%	2,303.5	0.18
End	All horizons	2,909.1	7%	1,832.3	0.54

reliable as a short-horizon state-transition model for aggregate factory movement: factory-level backlog remains within 6–8% relative error at the end of the horizon, and cumulative drained throughput retains positive correlation across horizons, while daily and workstation-level timing remains substantially noisier.

Appendix E

Agent Layer: Routing Validation and Interface

This appendix reports the scripted validation suite used to assess the LLM-based agent as an orchestration layer. The purpose of the suite is limited: it validates whether representative prompts are routed to the expected backend tools, whether the required tool arguments are passed correctly, whether negative-control prompts avoid tool calls, and whether numerical statements in the final response are grounded in either the user prompt or the tool outputs. It is not a user study and does not measure decision quality, planner adoption, or scheduling optimality.

E.1 Routing and Grounding Validation

In the suite each scenario specifies the prompt, optional dialogue history, expected tools, required arguments, forbidden tools, and whether numeric grounding is required. Project-planning scenarios are fixed to a pre-defined project at the latest day in the database shop-floor snapshot so that the routing behaviour is tested against a concrete point-in-time planning context.

Table E.1: Scenario coverage of the scripted agent routing and grounding validation suite.

Category	Scenarios	Expected routing behaviour
Manufacturing duration prediction	4	Duration prediction, duration what-if, and open-order tools
Project planning	4	Assign recommended start dates for orders in project
Commodity disruption monitoring	4	Appropriate tools for nickel and copper explanation, chart, and episode prompts
Cross-component and negative controls	3	Multi-tool routing where required, and no tool call for conversational or conceptual prompts
Overall	15	Representative orchestration, grounding, and no-tool behaviour

The evaluator records four category-level checks. Correct-tool rate measures whether the expected backend service was invoked. Valid-argument rate checks whether key identifiers and context fields, such as the manufacturing-order number, were passed correctly. Grounded-response

rate checks whether the agent introduced unsupported numerical claims in its final response. Task success requires tool selection, argument validation, forbidden-tool checks, no-tool behaviour where applicable, grounding, and backend availability to pass simultaneously.

Table E.2: Results of the scripted agent routing and grounding validation suite.

Category	Scenarios	Correct tool	Valid arguments	Grounded response	Task success
Manufacturing duration prediction	4	100.0%	100.0%	100.0%	100.0%
Project planning	4	100.0%	100.0%	100.0%	100.0%
Commodity disruption monitoring	4	100.0%	100.0%	100.0%	100.0%
Cross-component and negative controls	3	100.0%	100.0%	100.0%	100.0%
Overall	15	100.0%	100.0%	100.0%	100.0%

These results support only the narrow architectural claim that the implemented agent can satisfy its routing and grounding contract over the scripted scenario set. The perfect score should not be interpreted as evidence that the conversational interface improves planning or procurement decisions in practice, since that would require comparison against user decisions, planner workflows, or operational outcomes.

E.2 Agent Interface Walkthrough

The figures in this appendix illustrate the deployed conversational interface on a few representative interactions. They are included as a qualitative illustration of the implemented system rather than as evidence of decision quality: as set out in Section 5.3, the agent layer is assessed through a narrow routing and grounding check together with an informal, formative evaluation, rather than a controlled user study. All numerical values shown in the responses originate from the tool backends rather than the LLM's parametric knowledge, in keeping with the grounding contract described in Section 4.5; order and customer identifiers are anonymised where necessary.

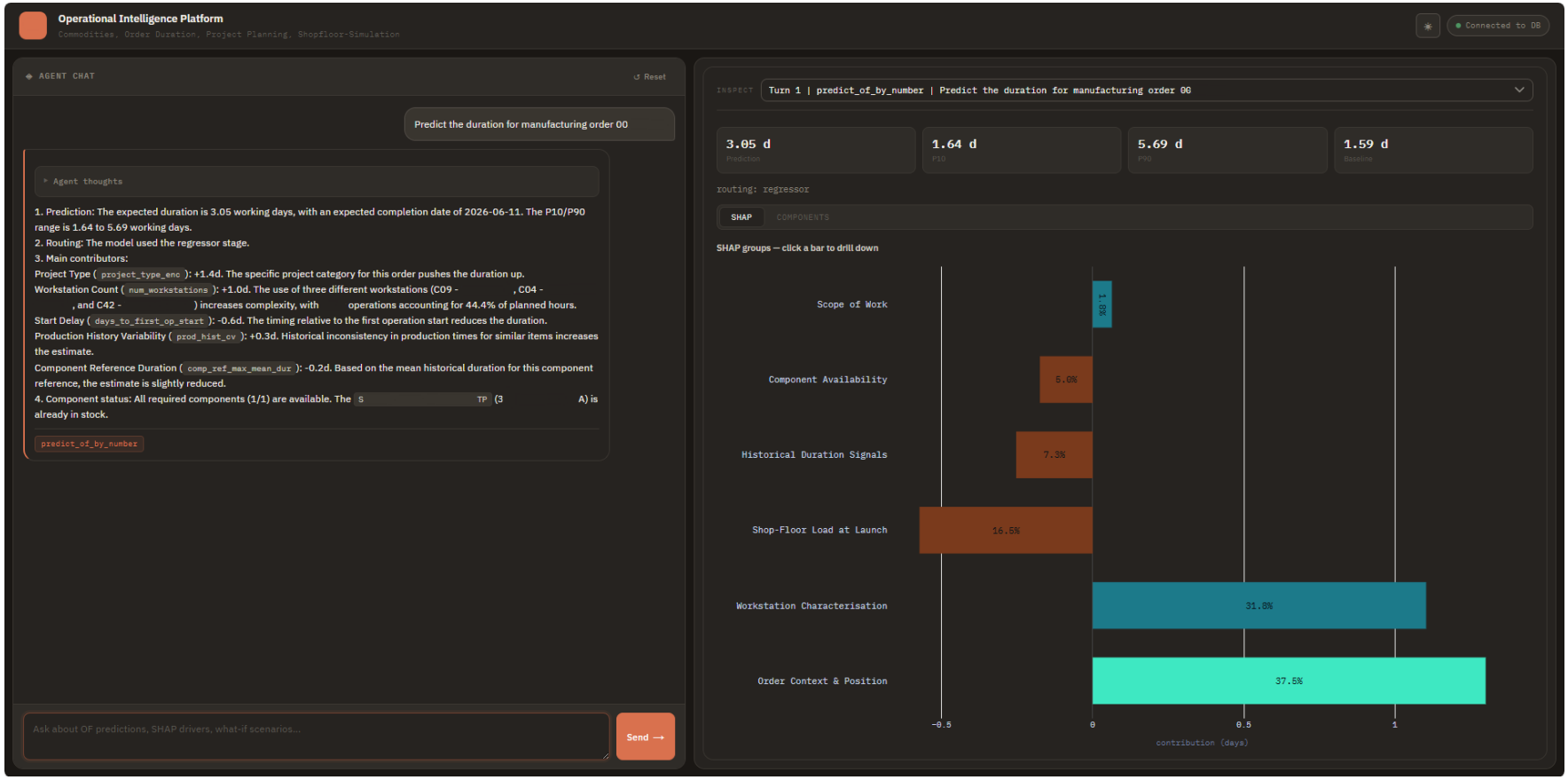

Figure E.1: Single manufacturing-order query. The agent returns the point duration estimate with its P10–P90 uncertainty interval and a natural-language breakdown of the dominant SHAP drivers, translating feature attributions into plain language rather than presenting raw importance scores.

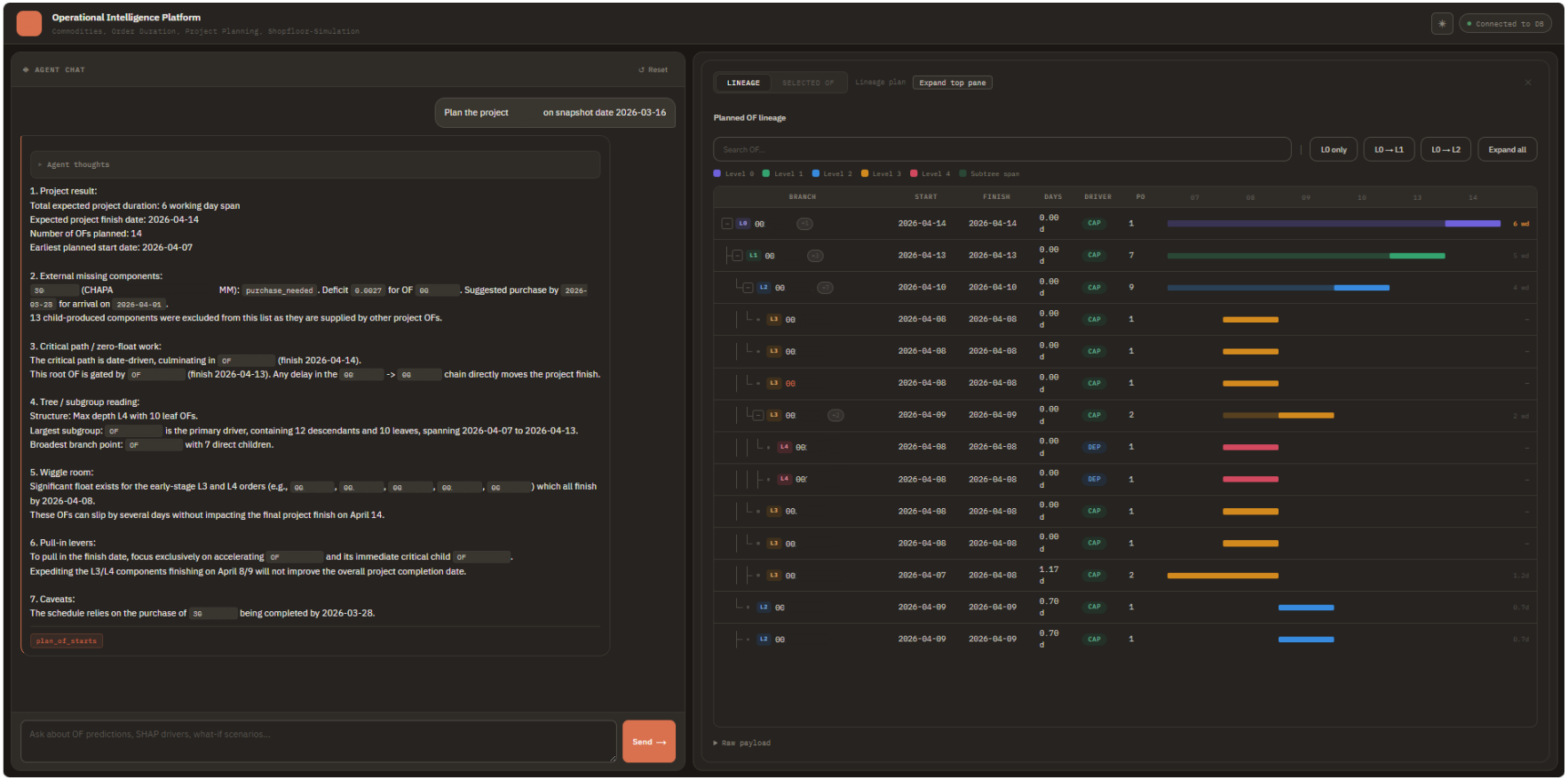

Figure E.2: Project-planning query. The agent invokes the planning tool and reports the recommended start sequence and expected finish for the project's manufacturing orders, together with the constraint diagnostics surfaced from the simulation backend.

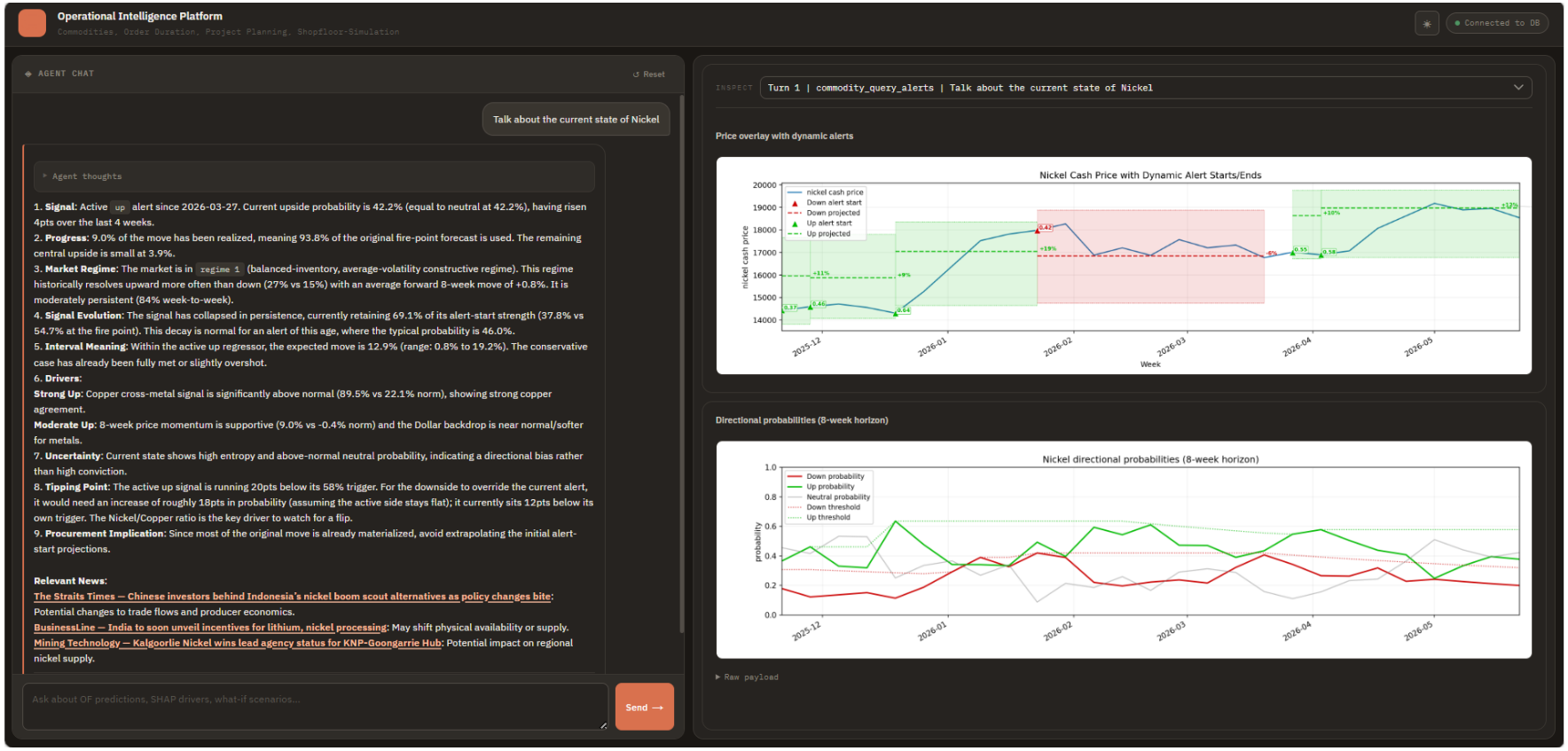

Figure E.3: Commodity disruption query. The agent reports the current alert status and renders the price-overlay chart produced by the commodity backend, illustrating how tool-generated visual artifacts are surfaced alongside the textual response.