

# AI Application Operations - A Socio-Technical Framework for Data-driven Organizations

Daniel Jönsson<sup>1</sup>, Mattias Tiger<sup>1</sup>, Stefan Ekberg<sup>2</sup>, Daniel Jakobsson<sup>3</sup>, Mattias Jonhede<sup>4</sup>, and Fredrik Viksten<sup>1</sup>

<sup>1</sup>Linköping University

<sup>2</sup>AI Sweden

<sup>3</sup>Swedish Transport Administration

<sup>4</sup>Volvo Group

December 15, 2025

## Abstract

We outline a comprehensive framework for artificial intelligence (AI) Application Operations (AI App Ops), based on real-world experiences from diverse organizations. Data-driven projects pose additional challenges to organizations due to their dependency on data across the development cycle. To aid organizations in dealing with these challenges, we present a framework outlining the main steps and roles involved in going from idea to production for data-driven solutions. The data dependency of these projects entails additional requirements on continuous monitoring and feedback, as deviations can emerge in any process step. Therefore, the framework embeds monitoring not merely as a safeguard, but as a unifying feedback mechanism that drives continuous improvement, compliance, and sustained value realization—anchored in both statistical and formal assurance methods that extend runtime verification concepts from safety-critical AI to organizational operations. The proposed framework is structured across core technical processes and supporting services to guide both new initiatives and maturing AI programs.

## 1 Introduction

AI systems are increasingly being deployed in critical domains where transparency, safety, and performance must be maintained across the full model lifecycle. To support sustainable and trustworthy AI adoption, organizations need clearly structured processes that span data, model development, deployment, and operations. This paper presents a framework for AI Application Operations, identifying common challenges and effective practices at each stage, with a particular emphasis on how continuous monitoring bridges the technical and organizational dimensions of AI practice. The framework has been developed based on expert interviews, workshops, and surveys within a large consortium of companies<sup>1</sup>, government agencies<sup>2</sup>, and research institutes<sup>3</sup> transforming into, or working with, data-driven organizations. Thus, it integrates perspectives from industry, government, and academia to facilitate alignment across organizational boundaries.

**Scope and contribution.** This white paper consolidates the collective experience of a multi-stakeholder consortium spanning public agencies, private companies, and research institutions. It is written as a *practitioner–research synthesis*: bridging empirical observations from real AI deployments with concepts from academic and industrial literature on machine learning operations (MLOps), data readiness, and AI governance. Whereas most prior works focus either on technical MLOps pipelines or isolated data-preparation phases, this paper proposes a coherent, end-to-end framework that spans the full lifecycle from idea to sustained value in production. In this respect, the work extends the canonical MLOps perspective by explicitly including (i) the **application and usage** stages—how models are integrated into real systems and continuously assessed in use—and (ii) the **value-driven meta-process** that links technical observability to business, ethical, and regulatory outcomes.

---

<sup>1</sup>Aixia, IBM, NetApp, Predli, Proact IT Sweden, Stormgrid, Volvo Group, Red Hat, Hopsworks

<sup>2</sup>Swedish Traffic Agency, Sahlgrenska University Hospital, Swedish Tax Agency, Statistics Sweden (SCB), Region Halland and Västra Götaland regional health care providers

<sup>3</sup>Linköping University, Research Institutes of Sweden (RISE), Santa Anna IT Research Institute

The white paper therefore aims to serve three audiences simultaneously:

- *Practitioners*, who can adopt the described processes as templates for reliable and compliant AI delivery;
- *Researchers*, who can view the framework as an empirical extension of existing MLOps and data-readiness theories; and
- *Policy makers and organizational leaders*, who can recognize how operational practices implement the principles of trustworthy and value-aligned AI.

The result is an explicitly *value-driven framework*; every technical activity from data curation to application integration is anchored in a declared *value hypothesis*<sup>4</sup> and advanced only when evidence supports that hypothesis, see Figure 1. Concretely, teams start by stating the intended outcomes (e.g., cost reduction, error avoidance, service-level improvements, user satisfaction), the proxies that can be measured early, and the guardrails (safety, legal, ethical) that must hold for the value to be deployable. Delivery then proceeds in stages (proof-of-concept → MVP → rollout), each with pre-defined thresholds that gate the next step. Operations maintain and increase value through continuous monitoring, analysis, and change—feeding downstream signals back into upstream work.

This stance both contextualizes and *constrains* process choices. For example, if downstream value depends on safe human triage, the model must provide calibrated uncertainty or abstain (“I don’t know”) under distributional shift. If value must be *defensible* in regulated contexts, explainability, lineage, and auditability become non-negotiable non-functional requirements. Likewise, instrumentation for value (business KPIs, task metrics, and safety indicators) is designed *before* building, so that experiments can be judged against the hypothesis.

The framework applies to both predictive and generative AI. Predictive systems typically articulate value in terms of decision quality and workload shaping (e.g., avoided false negatives, time-to-resolution), with calibration and cost-sensitive metrics tied to business outcomes. Generative systems articulate value in terms of task completion and quality (e.g., deflection rate, handling time, citation fidelity, refusal behavior), with policy and safety guardrails integrated into evaluation and serving. In both cases, value realization depends on the *application*—how predictions or generations change real workflows—not solely on model accuracy nor precision.

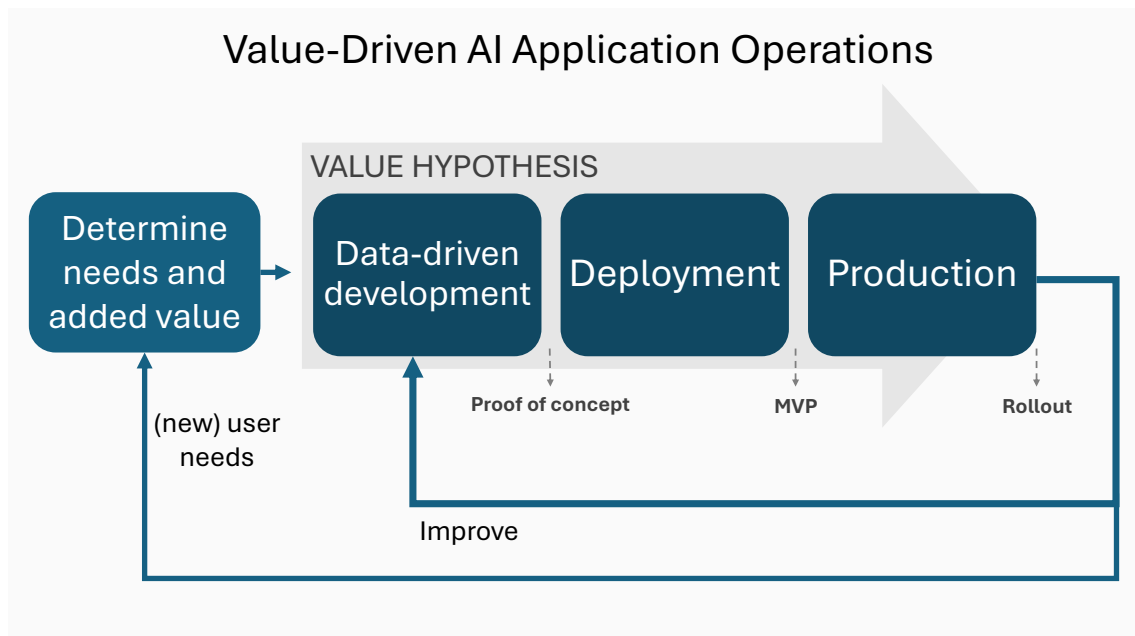


Figure 1: Illustration of the process for value-driven AI application operations, starting with determining business needs and finally using production-generated data for developing new applications or improving the existing ones. The process is inspired by the phases of CRISP-DM[1], but extends them to also take the feedback from deployed operations into account.

<sup>4</sup>For clarity, all key terms, abbreviations and concepts used throughout this paper are defined in the [Glossary](#) at the end.

## 2 Related Work

Machine Learning Operations (MLOps) has emerged as the principal framework for managing the lifecycle of machine learning models. Google’s *Practitioners Guide to MLOps* [2] and subsequent definitions such as Kreuzberger et al. [3] describe standardized processes and capabilities—experiment tracking, model training, deployment, and monitoring—aimed at reliable automation and reproducibility. These frameworks typically emphasize **automation, scalability, and reproducibility** but focus primarily on the model and infrastructure layers. Our work builds directly on this foundation but extends the scope beyond model deployment to include the surrounding *application, value realization, and governance* layers of AI systems.

Several complementary strands of literature address adjacent gaps. **Data readiness and data-centric AI** have gained renewed attention with works such as Lawrence’s *Data Readiness Levels* [4], the *Data Readiness Report* [5], and the recent extensions by Tiger et al. [6] and Jakubik et al. [7]. These emphasize systematic data preparation and quality documentation as prerequisites for reliable modeling. AIAppOps incorporates these insights by structuring the *Data Process* into model-agnostic curation and model-specific adaptation stages, thereby operationalizing data readiness within the continuous lifecycle.

A growing body of work examines **monitoring and maintenance** of deployed models. Naveed et al.’s multivocal review [8] shows that most monitoring practices still focus narrowly on technical metrics (e.g., latency, accuracy, drift), while aspects such as fairness, privacy, and business impact remain underrepresented. At the theoretical end, Tiger’s work on safety-aware autonomous systems [9, 10] formalizes monitoring as runtime verification under uncertainty, introducing Probabilistic Signal Temporal Logic (ProbSTL)<sup>5</sup> as a method for reasoning over probabilistic events and temporal guarantees. These contributions extend monitoring beyond metric collection to formal assurance, a concept that AIAppOps generalizes into socio-technical observability across the full AI lifecycle. The AIAppOps framework explicitly addresses these gaps by linking technical observability to value and trustworthiness indicators, ensuring that *data, model, inference, and application* are monitored jointly.

Finally, recent policy developments, particularly the *EU Artificial Intelligence Act* [11], stress continuous post-deployment monitoring, documentation, and human oversight for high-risk AI systems. The governance and compliance mechanisms embedded throughout AIAppOps provide a practical way to meet these regulatory expectations.

In summary, AIAppOps extends MLOps and related data-centric frameworks by introducing:

1. a value-driven meta-process aligning technical and organizational outcomes,
2. explicit processes for application integration and real-world usage, and
3. cross-cutting governance ensuring lawful, ethical, and sustainable operation.

## 3 Method

The core of our framework follows previous works on machine learning operations (MLOps) [1, 3], but takes a wider perspective by integrating value-driven and governance-oriented processes that extend beyond the traditional model lifecycle.

### 3.1 AIAppOps as an Extension of MLOps

MLOps provides the foundational practices for automating the development, deployment, and monitoring of machine learning models [3, 2]. However, most MLOps frameworks stop at the point where a model is deployed and technically monitored, e.g., [12, 3]. AI Application Operations (AIAppOps) extends this lifecycle in two essential dimensions:

- **Downstream extension:** adding the *Application* and *Usage* phases, which address how models are integrated into software systems, user workflows, and decision processes, and how value is measured and maintained in real-world operation.

---

<sup>5</sup>ProbSTL (Probabilistic Signal Temporal Logic) formalizes uncertainty-aware runtime monitoring by expressing probabilistic temporal constraints over continuous signals. Developed by Tiger [9, 10], it provides the mathematical underpinning for AIAppOps’ highest maturity level of monitoring.

- **Cross-cutting extension:** embedding *value alignment, trustworthiness, and compliance* as continuous feedback loops across all phases—from data to application—rather than as external audits.

In contrast to purely model-centric approaches, AIAppOps defines operational responsibilities for application developers, business owners, domain experts, and legal officers alongside data scientists and ML engineers. This broadens the scope of MLOps into a *socio-technical framework* that ensures not only that models are deployed correctly, but that AI applications continue to deliver intended outcomes safely, lawfully, and sustainably.

### 3.2 Workshops

We performed a series of workshops with organizations at various AI maturity levels, please refer to [subsection 4.1](#) for maturity level assessment. Each workshop started with a description of the organization’s current process with concrete examples of past and ongoing data-driven projects. The process description session was followed by discussions about the challenges they faced in their current stage and which challenges they saw in the near future. We then adapted and aligned our process description to capture best practices from both theirs and previously interviewed organization’s data-driven processes. This allowed us to come up with a process description that captured both best practices described in previous literature [3, 13] and the current workflow of the interviewed organizations.

After the initial workshops with the organizations, we discussed the resulting process description to ensure that the abstraction levels were appropriate and clear. Finally, we presented the process description to a larger number of new organizations to ensure that it captured and covered the needs in diverse sets of AI applications and organizations. Comments and discussions after the presentation were integrated into the process description, resulting in the AI Application Operations process presented in the following section.

## 4 AI Application Operations

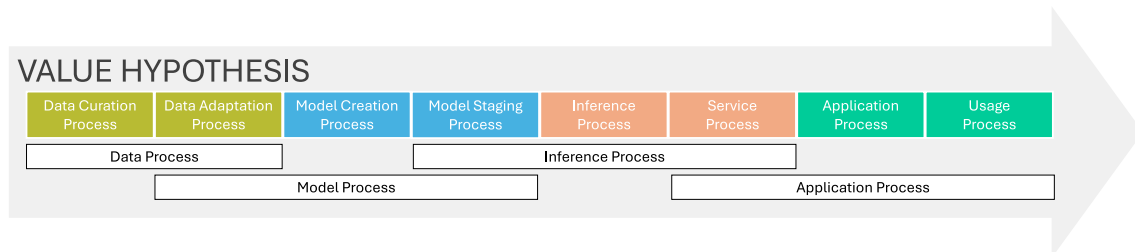


Figure 2: A layered view of the processes in the AI application life-cycle. In addition to the traditional data, model, and inference (serving) processes, we also highlight the application and its usage due to their importance for the value hypothesis. All processes and process levels are iterative and continuous.

We divide the AI application life-cycle into four major processes illustrated in [Figure 2](#): Data, Model, Inference, and Application. Here, the data process makes data ready for model development, the model process transforms curated data into models, the inference process deals with reliably serving inferences based on requests to models, and the application process deals with the more traditional development operations using data-driven services. The major processes are often not completely separated, which we indicate through the overlap between the processes with the in-between steps: data adaptation, model staging, and service provision. Furthermore, the overlap also highlights the need for inter-team communication and overlapping roles between the steps of the AI application operations.

Here, data adaptation involves model-dependent transformations, model staging validates models for real-world usage, and service provisioning orchestrates chains of inferences as a service to the application.

The linear depiction of the process in [Figure 2](#) reflects that all steps are required for a deployed solution. However, it should not be interpreted as one-directional work-flow. In fact, it is common with iterations between the processes. For example, to acquire and adapt new data based on identified needs of the application. Across all processes, support functions such as monitoring,

improvement cycles, and compliance are crucial to maintaining application integrity over time. The [subsection 4.7](#) details how monitoring operationalizes this continuity by integrating data, model, and application observability into a single feedback loop.

## 4.1 Organizational Maturity and Its Impact on AI Operational Processes

Few organizations begin their AI journey with a fully developed process that encompasses the entire AI application life-cycle. Instead, most start with isolated projects that serve as learning experiences and stepping stones. These initial efforts help build the necessary competencies and process components required to eventually manage AI applications systematically and at scale.

Once one or a few AI applications are successfully deployed in production, it becomes meaningful to strive for a more complete and integrated operational process. As a reference point, we provide a distilled process map (see [Figure 3](#)) that illustrates how a mature organization may manage AI application operations. This process includes not only the delivery of AI-powered services but also the organizational structures and feedback mechanisms that enable continuous learning, monitoring, and reuse of data and models across applications.

It is important to emphasize that implementing the entire AI application operations process from the outset is rarely feasible nor advisable. The ability to manage the full AI life-cycle—including continuous deployment, monitoring, and feedback loops—depends heavily on the organization’s operational maturity.

### 4.1.1 Assessing Organizational Maturity

Before starting to transform the organization to AI application operations one should assess the current maturity of the organization [14]. There are several scales that can guide organizations in evaluating where they stand in their journey from ad-hoc experimentation to enterprise-wide AI operations, e.g., [15, 16]. Here, we use the 5-level MLOps maturity scale defined by Microsoft [16]. These can be briefly summarized as:

- **Level 0 - No MLOps:** Manual, script-driven processes with no tracking of model performance. Difficult to manage the full ML lifecycle, and releases are painful.
- **Level 1 - DevOps, No MLOps:** Application code is managed with automated builds and tests, but model training remains manual. Feedback from production is limited.
- **Level 2 - Automated Training:** Fully managed training environment with centralized model tracking and reproducible results. Manual but low-friction releases.
- **Level 3 - Automated Model Deployment:** Automatic, low-friction releases with full traceability and integrated A/B testing. Entire environment managed from training to production.
- **Level 4 - Full MLOps:** Completely automated system with continuous model retraining, verbose centralized metrics, and automatic improvements approaching zero-downtime operations.

Once you have assessed your organizational maturity, you can use it to guide the implementation of AI application operations.

### 4.1.2 Considerations Based on Maturity Level

To successfully implement the entire AI application operations into the organization, one should adopt a phased approach. Attempting to implement a full-scale AI operations framework prematurely can lead to inefficiencies and frustration. Instead, organizations should consider the following depending on their organizational ML maturity level:

- **Level 0:** Start with proof of concept projects to build experience, stress-test the organization, and identify gaps. At this stage, it’s crucial to avoid over-engineering and instead prioritize learning and experimentation. You might not reach the application process, but gain organizational experience.

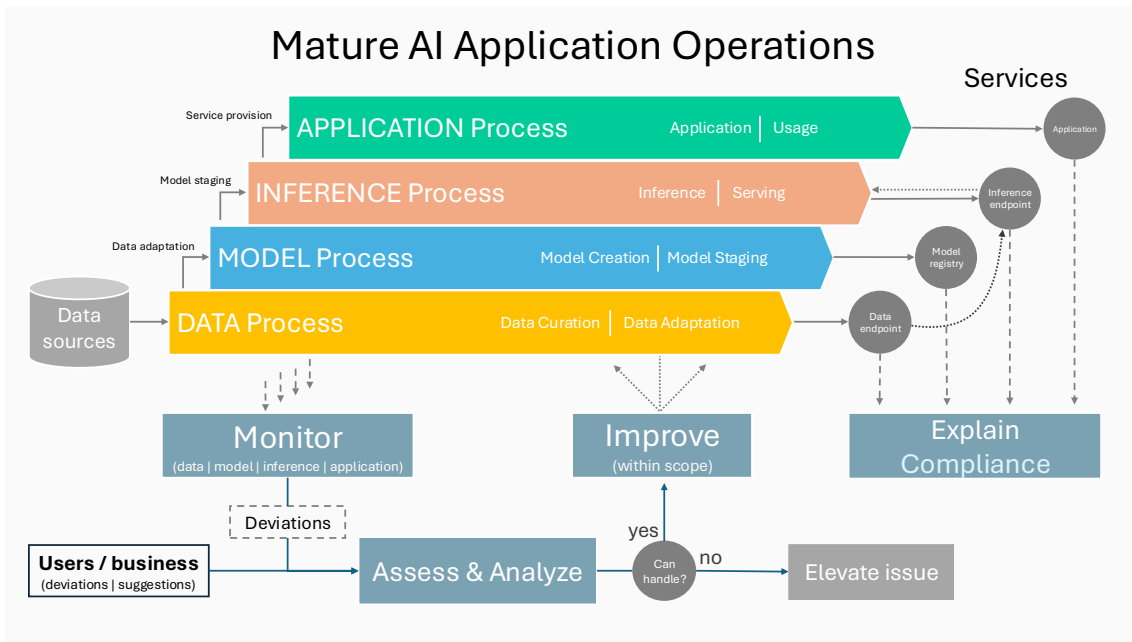


Figure 3: High-level map of AI application operations for an organization with high machine learning operations maturity. The four core operational processes—Data, Model, Inference, and Application—each produce composable endpoints used in downstream services. Monitoring and feedback mechanisms support continuous improvement, while compliance and explainability services ensure operational trustworthiness. The framework supports both proactive and reactive governance by linking technical observability to business-facing outcomes, and by embedding continuous monitoring as the connective tissue between all lifecycle processes.

- **Level 1:** Go beyond pilot projects, i.e., MVPs and rollouts so that the organization obtains experience of the entire AI application life-cycle. Focus on building core capabilities. This includes establishing basic data governance, reproducible training pipelines, and initial monitoring mechanisms.
- **Level 2:** With automated training in place, organizations can begin to standardize processes and introduce centralized model tracking. This is a good point to start planning for scalable deployment strategies and to define roles and responsibilities for AI operations.
- **Level 3:** Align organizational structures and roles to support AI operations. Organizations can now implement automated deployment pipelines and integrate feedback loops. This enables faster iteration and more reliable performance in production environments. Cross-functional collaboration becomes essential to manage dependencies between the data engineering, data science, ML engineering, and the application teams.
- **Level 4:** Foster a culture of data-driven decision-making and continuous improvement. Here, the focus shifts to feedback mechanisms, optimizing performance, minimizing downtime, and leveraging shared data and models across applications to maximize value.

## 4.2 Roles

The delineation of roles in AIAppOps intentionally diverges from the more infrastructure-centric characterizations found in major MLOps frameworks (e.g., Google’s *Practitioners Guide to MLOps* [2]) and from the narrow technical abstractions in many industrial “ML team topologies.” In contrast to frameworks that primarily emphasize the automation of pipelines and handoffs between data scientists and ML engineers, the AIAppOps role model explicitly integrates the *application*, *governance*, and *value-realization* perspectives that determine whether machine learning systems sustain value and compliance in production.

Compared to the canonical MLOps lifecycle [2, 3], where the primary boundary lies between data engineering and ML engineering, we introduce three critical extensions. First, we incorporate **application engineers**, who translate model capabilities into user-facing and decision-support systems, thus closing the loop from prediction to realized value. Second, we include **governance and legal functions** as first-class participants, ensuring that compliance, ethics, and accountability are built into daily operations instead of appended as external audits. This prevents what practitioners often call the “compliance-last bottleneck,” where legal review occurs only at deployment time, leading to costly rework or operational risk. Third, we integrate **business roles** into the whole process to ensure that value hypotheses, success metrics, and organizational constraints continuously shape technical choices rather than being evaluated post hoc.

A pre-emptive clarification is warranted for readers accustomed to role frameworks such as those in DataOps or DevOps. While these frameworks also assign cross-functional ownership, their boundaries end at data pipelines or software artifacts. AIAppOps extends this logic to the *decision layer*—where data, models, and human oversight intersect—and therefore requires inclusion of roles (e.g., **DME**, **LG**, **BA**, **PM**) often treated as peripheral in MLOps literature. By formalizing these as operational participants, the framework reconciles technical excellence with lawful, ethical, and value-aligned operation, in line with the principles of trustworthy AI [17, 11].

Here, we briefly describe the different roles and later map them into the four main processes of AI application operations.

- DE Data Engineer** Builds and maintains the foundational data pipelines and platforms (like data lakes or feature stores) that make data discoverable, reliable, versioned, and available for consumption.
- GO Governance Officer / Data Steward** Ensures data quality, compliance, and proper data management practices. Is responsible for achieving and maintaining high data readiness (explainable and interpretable data, i.e. information, by man and machine) falls upon this role. They define data standards, monitor data usage, and enforce policies related to privacy, security, and regulatory requirements. Their role is crucial in maintaining trust and accountability in data.
- DS Data Scientist** Applies statistical analysis, machine learning, and domain knowledge to extract insights from data and data readiness. They need to easily find, understand, and experiment with different datasets and features to build and train models, relying on data catalogs and version control.
- MLE Machine Learning Engineer** Specializes in deploying and maintaining models at scale. They work closely with data scientists to productionize models, optimize performance, and ensure models are robust, reproducible, and integrated into software systems or products.
- AE Application Engineers** Designs, develops, and maintains the software and user-facing systems through which AI capabilities create value. Application Engineers encompass software developers, DevOps specialists, and UX or interaction designers working together to integrate AI services into real applications. Their work spans backend and frontend components—APIs, orchestration layers, services, and user interfaces—that connect inference endpoints with other business logic and human workflows. They also manage infrastructure-as-code, automated testing, and CI/CD pipelines to ensure secure, reliable, and continuously evolving operations.
- BA Business Analyst** Collaborates with stakeholders to understand the business needs and translate them to requirements and success metrics (e.g., KPIs). Works together with data scientists to validate assumptions and interpret model results from a business perspective. Prepares documentation, dashboards, and reports to communicate insights and progress.

**PM Product Manager** Owns the product vision and ensures that the solution delivers value through defining business requirements, prioritizing features or use cases. Acts as a bridge between technical teams and business stakeholders. The role can also be referred to as product owner when situated closer to the customer.

**DME Domain Expert** Brings deep knowledge of the industry or specific business area. They help interpret data in context, validate findings, and ensure that data solutions align with real-world use cases. Their input is essential for data understanding, data curation, accurate modeling and effective decision-making.

**LG Legal** Ensures that all data practices comply with applicable laws, regulations (e.g., GDPR, HIPAA, CCPA), and contractual obligations. They review data usage policies, assess legal risks, guide consent and data-sharing frameworks, and work with technical and governance teams to mitigate liabilities. Their input is critical in safeguarding the organization from legal and reputational harm.

### 4.3 Data process → Data endpoints



The data process begins with sourcing and preparing raw data, converting it into formats usable by machine learning models. This corresponds to raising the dataset’s *data readiness level* [4], from raw and inaccessible data to curated, task-ready datasets documented via a Data Readiness Report [5].

We divide the process into two sub-processes: *data curation*, which includes model-agnostic steps focusing on data quality, and *data adaptation*, which transforms the data according to modeling assumptions. The reason for this separation from a process perspective is two-fold. First, it clarifies the role separation within the data process where data engineers are primarily involved in data curation, while a data scientist adapt and experiment for model-specific needs. Second, it highlights the complexity caused by the fact that some data processing is performed closer to, or even during, the model development. A typical such example can be data augmentation. In the end, the process should provide versioned data endpoints. These endpoints expose the possibly transformed data (feature store) for model training, inference, and monitoring.

#### 4.3.1 Data Curation → Clean data

Data Curation refers to the systematic management and refinement of data to ensure it is fit for purpose. It begins with ensuring access to data, followed by systematic steps to refine it. The ultimate goal is to produce a high-quality, well-documented, and trustworthy dataset that is not only fit for its initial purpose but is also discoverable, reusable, and reproducible for future applications. The key steps involved are:

- **Data Accessibility:** The ability to retrieve and legally use the data required to build, train, and deploy a model. Data can be siloed in different departments where its existence is passed down through ”tribal knowledge”. To find something, you have to ask multiple people, often leading to dead ends. To mitigate this, the organization can use a centralized searchable repository containing metadata about all available datasets. Once found, it is necessary to investigate if the data is *allowed* to be used or if there are ways to allow the use of the data. It must also be possible to retrieve the data with reasonable efficiency. Accessibility can be hindered by need for manual handling, e.g., receiving files via email, the size of the data, or the way it is stored.
- **Data Understanding** involves both initial investigation of the ability of the accessible data to solve the problem and as well as understanding its quality. This includes both quantitative analysis, e.g., by computing descriptive statistics (mean, median, standard deviation, counts, frequencies etc.) to summarize feature properties, and qualitative analysis, which relies on data visualization (e.g., histograms, scatter plots, box plots) to uncover patterns, distributions, data imbalances, correlations, and potential anomalies. This phase is crucial for assessing the data’s suitability for the project goals, identifying data quality issues (like

missing values or outliers), and forming initial hypotheses that will guide the subsequent data preparation and modeling stages.

- **Data Alignment** concerns the explicit alignment between the data used for modeling and the *normative intent* of the application. At this stage, teams must decide whether the model should reflect patterns observed in the world as it *is*—including historical biases, structural inequalities, and past decision practices—or the world as it *ought to be*, as defined by legal, ethical, and organizational principles.

Data that faithfully mirrors historical reality can be appropriate and even necessary for certain purposes, such as descriptive analytics, counterfactual reasoning, policy simulation, or synthetic data generation. In contrast, decision-support and automated decision-making systems typically require data that is aligned with normative constraints, including non-discrimination, proportionality, and regulatory compliance. In such cases, historical biases present in the data must be identified, documented, and actively mitigated rather than passively learned.

Data alignment therefore involves making value-laden choices explicit: selecting target populations, defining protected attributes and acceptable proxies, determining which patterns are permissible to learn, and deciding where rebalancing, reweighting, or data augmentation is required. These decisions are inseparable from fairness, accountability, and explainability considerations, and must be documented as part of the dataset’s lineage and justification.

- **Data Cleaning** involves detecting and resolving errors and inconsistencies in data to enhance its quality. This includes correcting structural errors, standardizing disparate formats, and removing duplicate or irrelevant observations. It can also include handling of uncertain data (e.g., through removal or by adding uncertainty descriptions) or missing values (e.g., through imputation or removal), but this can also be addressed in the data adaptation step as it can be model-dependent. The primary goal is to produce a trustworthy and reliable dataset for robust downstream analysis and modeling.
- **Data Validation** ensures that the data follows its assumptions. It often involves programmatic checks the data against a predefined schema and a set of constraints derived from the source, domain knowledge, or a reference dataset. These checks typically include verifying data types and column presence, ensuring values fall within expected ranges, and confirming that key statistical properties (e.g., mean, variance, class distributions) have not significantly deviated. The more that is known about the domain, the more precise in-vs-out-of-distribution validation checks can be used.

### 4.3.2 Data Adaptation → Model-specific data

Within the model operations pipeline, the Data Adaptation stage is responsible for the transformation of curated data into a specialized format optimized for a target model. These transformations can typically not be reused across different types of models or model types. This generally involves:

- **Data Transformation** within the data curation step performs model-agnostic preprocessing steps that are generally beneficial across a wide variety of machine learning algorithms. They address common data quality issues rather than catering to the specific architectural needs of one model. These transformations are considered “agnostic” because they improve the numerical stability, interpretability, and performance of most models, from simple linear regressions to complex neural networks. A summary of common such transformations is provided in [Table 1](#),
- **Model-Specific Feature Engineering** creates or transforms features based on the inductive biases of the model architecture (e.g., tokenization for Transformers, spatial transformations for CNNs).
- **Labeling** prepares the data for supervised machine learning data points are assigned an informative tag or output value. This label represents the “correct answer” or “ground truth” that the model is expected to predict. The quality, accuracy, and consistency of these labels are paramount, as they directly define the objective function for model training and establish the gold standard for performance evaluation.
- **Data Augmentation** generates variations of the data to enhance training set diversity. Augmentations are often performed during the training process. Here, the versioning can

Table 1: Summary of common model-agnostic data transformations.

| Transformation Type             | What It Does  | Primary Benefit  |
|---------------------------------|---|--|
| <b>Feature Scaling</b>          | Puts numeric features on a common scale.                  | Improves convergence for gradient-based models; essential for distance-based models. |
| <b>Categorical Encoding</b>     | Converts text labels into numbers.                        | Makes data compatible with virtually all ML algorithms.                              |
| <b>Outlier removal</b>          | Removes or caps outliers to reduce their impact on models | Improve model performance for common cases.  |
| <b>Distributional Transform</b> | Reduces skewness in data.                                 | Stabilizes variance and helps models meet statistical assumptions.                   |
| <b>Discretization/binning</b>   | Groups continuous values into discrete bins.              | Can capture non-linear patterns in a simpler, model-agnostic way.                    |

be more related to the types and variability of the augmentations rather than the resulting augmentations. The versioning becomes a trade-off between exact reproducibility and storage or computational resources.

The output of this stage is a versioned set of data artifacts, managed in a model-specific feature store, which guarantees data lineage and enables the reproduction of training and inference environments.

#### 4.3.3 Data engineering team DE GO + (DME LG BA)

This process mainly involve a smaller subset of the team. Data Engineer and data steward make data discoverable, compliant, high-quality, and versioned. Communication with legal is necessary to clarify uncertainties and obstacles around data usage. The data scientist explores, validates, and adapts data (features, labels) with domain expert input to ensure domain fidelity. Project manager and business analyst define value hypotheses and data SLAs (freshness, coverage) and align curation/adaptation to downstream needs.

## Data process value connection

Data work creates leading indicators for value realization by ensuring that downstream decisions are made on lawful, relevant, timely, and representative data. Typical indicators include dataset coverage and freshness, label quality and agreement, and baseline drift monitors (schema, distribution, and concept proxies). These are tied to application-level hypotheses (e.g., “freshness < 24h is required to reduce incident handling time by 15%”), making data Service Level Agreements (SLAs) directly consequential for the value hypothesis.

## 4.4 Model process → Model registry



The model process transforms curated data into trained and deployable AI artifacts. This phase aligns with the MLOps core pipeline [3, 2], but here is extended with governance and staged validation before deployment. It encompasses the design, training, validation, and operational readiness of models, ensuring that learned representations faithfully encode the intended patterns and behaviors. This process is central to value creation in AI projects and must accommodate experimentation, iteration, and evolution over time. We divide the model process into two sub-processes: *Model creation*, where data becomes a trained model, and *Model staging*, where the model is prepared for real-world use through evaluation and controlled deployment.

This phase is tightly coupled with the data adaptation process. Since model requirements and architectures evolve throughout experimentation, the features and transformations in data

adaptation must remain flexible and co-developed with modeling efforts. This dynamic interplay ensures model fidelity, reproducibility, and performance in context.

#### 4.4.1 Model Creation → Trained model

Model creation focuses on converting data into a trained, versioned, and well-documented model artifact. It involves iterative experimentation to identify the most suitable modeling paradigm and configuration, guided by both the structure of the data and the desired application behavior. Crucially, model objectives, loss functions, and evaluation criteria must remain consistent with the data alignment choices established during data curation, ensuring that optimization reinforces the intended normative behavior rather than merely amplifying historical patterns. Key activities within this phase typically include:

- **Model selection** involves choosing a modeling paradigm and concrete architecture that aligns with the structure of the data, the nature of the task, and operational constraints. Currently, gradient-boosted decision trees (e.g., XGBoost, LightGBM) are often chosen for structured tabular data, probabilistic machine learning (e.g., Gaussian Processes, causal inference models) for uncertainty- and intervention-aware inference in sensitive and critical domains, and deep learning architectures (e.g., Transformers, CNNs) for unstructured data like text, images, and video. Sequence models (graphical models) such as Conditional Random Fields remain relevant for structured prediction in domains like NLP, automatic control and bioinformatics, while ensemble models like Random Forests offer robustness and interpretability in settings with heterogeneous features. The selection process balances predictive performance with considerations such as training efficiency, interpretability, deployment latency, and maintainability over time.
- **Hyperparameter tuning** finds the optimal settings for the selected model or even the model itself [18]. For example by adjusting learning rates, regularization parameters, or model complexity controls. In probabilistic machine learning, hyperparameter tuning often overlaps with model selection itself, as adjusting priors or kernel functions effectively changes the model’s hypothesis space. Unless the model is very expensive to train, it is often trained during the hyperparameter tuning (the best trained model is picked at the end).
- **Pre-training** uses large, general-purpose datasets to initialize model parameters with transferable knowledge, reducing the amount of task-specific data needed later.
- **Fine-tuning** specializes a pre-trained model to a specific domain or task using targeted data, often enabling higher performance with less data resource usage.

*Model Creation provides:* A version-controlled model registry that tracks model lineage, training configuration, and evaluation metrics. This enables reproducibility, auditability, and structured experimentation.

#### 4.4.2 Model Staging → Accessible validated model

Model staging serves as the bridge between experimental results and operational readiness. It prepares models for deployment through rigorous testing, real-world validation, and alignment with non-functional requirements such as latency, reliability, and compliance. Typical elements of this sub-process include:

- **A/B testing** performs empirical comparison of multiple models—or models against human decision-makers—in live or simulated environments to assess relative performance in business-relevant terms.
- **Shadow deployment** deploys a model in parallel with the current production system—without affecting outputs—to gather operational metrics and build trust before full integration.
- **Re-training and iteration** update models to account for data drift, concept shifts, or evolving business needs using monitored feedback or scheduled refresh cycles.

*Model Staging provides:* A validated deployment pipeline and model delivery framework, ensuring that trained models are production-ready, governed, and performance-assured under real conditions.

### 4.4.3 Data science team DS + (MLE DME)

Within the model process, the data scientist leads model selection, training, and evaluation; communication with the domain expert ensures valid behavior and failure modes. The ML engineer codifies experiments and works with the ML operations to ensure reproducibility, packaging, and registry lineage.

## Model process value connection

Model metrics are mapped to business outcomes before training. For predictive systems, calibration, cost-weighted error, and selective prediction (abstention/deferral) are linked to decision cost and safety thresholds. For generative systems, task success, factuality/groundedness, refusal and redaction rates, and toxicity/policy conformance are linked to user productivity and risk. The registry stores these mappings with lineage so that value regressions can be traced and explained.

## 4.5 Inference process → Inference endpoint



The inference process operationalizes trained models by integrating them into real-time or batch-serving environments. This phase transforms models from a theoretical artifact into a system component, reliably executing predictions under performance, availability, and compliance constraints. Industry frameworks typically treat this as the endpoint of MLOps [2]; AIAppOps continues beyond inference to encompass service orchestration, application integration, and user impact evaluation. It includes the infrastructure and logic needed to validate input data, route requests through appropriate inference paths, and optimize throughput. We divide this process into three sub-processes: *Model staging* (covered earlier), *Inference serving*, and *Service orchestration*.

### 4.5.1 Inference Serving → Live endpoints

Inference serving is where trained models are exposed through programmatic endpoints. It ensures that models can be accessed by other systems or users through interfaces optimized for latency, throughput, and security. This sub-process typically involves several key components that ensure the reliability and efficiency of executing model predictions in production environments:

- **Input validation:** Ensuring that requests conform to expected schemas and that dangerous or malformed inputs are rejected before reaching the model.
- **Batch and real-time handling** for supporting different invocation modes—e.g., batch processing for large datasets, or real-time APIs for instant decisions.
- **Scalability** by dynamically allocating resources (e.g., pods, containers, graphical processing units) based on request load using autoscaling policies.

*Inference Serving provides:* An inference endpoint with well-defined SLAs, suitable for integration with downstream services. It ensures model calls are reproducible, version-controlled, and observable.

### 4.5.2 Service Orchestration → Composable inference

In some deployments, a single prediction task may require a chain of model invocations, pre/post-processing, or additional business logic. Service orchestration builds such inference graphs—deterministic or agentic—using orchestration frameworks that support robustness, caching, and tracing. Robust service orchestration is built on a combination of architectural and runtime elements designed to ensure flexibility, control, and resilience:

- **Inference pipelines** for sequences of preprocessing, model calls, and postprocessing steps, possibly involving different modalities or models.

- **Endpoint selection** chooses which model version or instance to invoke based on metadata such as region, compliance, or A/B configuration.
- **Agentic inference orchestration** of LLM agents or graph-based runtime interpreters to decide which submodels to call and in what order, enabling adaptive workflows.
- **Resilience strategies:** Circuit breakers, fallbacks, and redundancy mechanisms to guarantee availability and degrade gracefully under fault.

*Service Orchestration provides:* A robust service layer on top of inference endpoints that supports composition, fault tolerance, and optimized throughput for diverse applications.

### 4.5.3 ML engineering team (MLE) + (AE PM)

The main work here is performed by the ML engineer, who provides performant, secure model serving (schemas, autoscaling, version routing). However, they must communicate with the application engineer team to ensure consistency with needs of the application. On the organizational level the project manager works with SLAs and cost targets.

#### Inference process value connection

Inference SLAs (latency, availability, cost-per-call) are framed as value proxies: if decisions arrive late or are intermittently unavailable, downstream value erodes. Orchestration introduces canaries and A/B configurations so that value hypotheses can be tested safely in production. Resilience strategies (circuit breakers, fallbacks to safer models or retrieval-only answers) are selected to *preserve value under fault* rather than merely maintain uptime.

## 4.6 AI application process → User-facing services



Building traditional applications involves a wide range of tasks covered by for example DevOps [19]. Here, we focus on the parts of the application development that are impacted by the use of a data-driven solution. The AI application process bridges model outputs with user-facing systems and business operations. This process ensures that predictions flow into real applications—such as dashboards, APIs, or embedded controls—while maintaining CI/CD best practices and aligning with KPIs and business goals & requirements. We divide this process into three sub-processes: *Service deployment*, *Application integration*, and *Downstream engagement*.

### 4.6.1 Service Deployment → Operational AI services

This sub-process focuses on deploying and maintaining the logic and infrastructure around AI services. It integrates inference pipelines into backend services that expose predictions via APIs, UIs, or event-driven systems. To operationalize AI services effectively, this phase integrates infrastructure, automation, and monitoring capabilities through components such as:

- **DevOps for AI services** builds and maintains CI/CD pipelines that validate and deploy service logic, infrastructure as code for reproducibility, and canary or shadow deployments for gradual rollout.
- **Monitoring and alerting:** Continuous observation of latency, throughput, error rates, and business metrics.
- **Compliance pipelines:** Automated checks to ensure that services using AI maintain required documentation, audit logs, and approval checkpoints.

*Service Deployment provides:* An AI-powered service layer with observable, maintainable, and policy-compliant interfaces.

#### 4.6.2 Application Integration → System-level access

Once the AI service is deployed and observable, it must be woven into the broader software and organizational ecosystem. This integration ensures that predictions reach users or systems at the right time, in the right format, and with the necessary context. Whether the service is consumed internally or externally, this phase is where AI begins to directly shape workflows, decisions, and products. Key aspects of integration typically include:

- **Interface design and exposure** through structuring the service contract — what inputs are accepted, what outputs are returned, and under what guarantees (e.g., latency, reliability, interpretability). Interfaces must remain stable and versioned over time to support downstream compatibility.
- **Embedding in applications** by connecting AI services to operational systems such as dashboards, case management tools, process automation engines, or end-user applications. This often requires adapting output formats, handling errors gracefully, and aligning responses with business needs.
- **Security and access governance** controls who can invoke AI services, how data flows are logged and monitored, and whether outputs meet domain-specific compliance requirements (e.g., access audits, rate limiting, encryption at rest and in transit).

*Application Integration provides:* An application endpoint that makes AI functionality available to users, applications, or automated agents.

#### 4.6.3 Downstream Engagement → Real-world impact

This final stage evaluates how well the deployed application supports real-world tasks and decision-making. It focuses on measuring outcomes, learning from user behavior, and closing the loop between model predictions and business results. To close the loop between prediction and business value, this phase emphasizes mechanisms for impact measurement and user-driven feedback:

- **Task performance analysis** measures whether model outputs result in improved efficiency, quality, or user satisfaction.
- **User feedback loops** captures corrections, exceptions, or suggestions from human users and feeding them back into training or retraining loops.
- **Key Performance Indicator (KPI) alignment** correlates application behavior with key business metrics (e.g., cost savings, error reduction, time-to-resolution).

*Downstream Engagement provides:* Real-world impact validation and continuous signals for improving models, services, or interfaces.

#### 4.6.4 Core application team AE DME BA PM GO LG

Here, the main work is performed by the application engineers to develop the AI-backed services and integrate inference into systems and UIs. However, this process may involve many of the roles that have been part of the project as both the data processing and model creation impact the usage within the application. Thus, it is important to either provide necessary documentation and/or allow for communication with the team members involved in the upstream tasks of processing the data and developing the models. The business analyst's role here is to finally measure the impact and build dashboards for monitoring metrics. The project manager steers the roadmap via KPIs.

### Application process value connection

Applications are where value is realized and observed. Core signals include adoption and override rates, time-in-state and rework, error and escalation patterns, and user satisfaction. These are wired back to inference, model, and data as structured feedback (labels, prompts/policies, features), enabling targeted improvements that move the same business KPIs that justified the project.

## 4.7 Monitoring → Value-drift tracking

Monitoring machine learning systems has long been recognized as a complex, multi-dimensional challenge [8]. Organizations often struggle to synthesize signals from data, models, infrastructure, and applications into a coherent operational picture. AI Application Operations (AIAppOps) addresses this challenge by integrating the monitoring of data drift, model behavior, and application-level KPIs into a unified feedback loop that ties technical observability to business outcomes. Through this integration, monitoring becomes not merely a safeguard, but a continuous process for ensuring that deployed AI systems remain reliable, safe, lawful, and valuable throughout their lifetime. Whereas traditional MLOps frameworks typically emphasize technical metrics such as model accuracy, latency, or data drift [2, 3], AIAppOps extends the scope of monitoring into a continuous socio-technical activity that links *technical observability* to *business value*, *human oversight*, and *compliance*. It turns telemetry into assurance by embedding measurement, explanation, and adaptation across all life-cycle stages.

### 4.7.1 Purpose and scope

Monitoring in AIAppOps pursues four intertwined goals. First, it safeguards the **effectiveness** of predictive and generative systems by tracking performance, calibration, and alignment over time. Second, it maintains **service quality** by ensuring that availability, latency, and cost adhere to agreed Service-Level Objectives (SLOs). Third, it reinforces **trustworthiness** through continuous checks on safety, fairness, privacy, and explainability under evolving conditions. Finally, it secures **value realization** by verifying that technical outcomes continue to produce the intended organizational or societal impact.

Recent reviews highlight that industrial monitoring often focuses narrowly on technical metrics, leaving fairness, interpretability, and business impact under-represented [8]. AIAppOps explicitly closes this gap by unifying these perspectives within one feedback loop.

### 4.7.2 Layers of observability: what to monitor

AIAppOps defines monitoring as a layered system of observability, with each layer corresponding to a critical dimension of the AI lifecycle—from raw data to realized value:

- **Data:** schema adherence, completeness, distributional and concept drift, and freshness; derived from data readiness concepts [4, 6].
- **Model:** performance, calibration, uncertainty, out-of-distribution (OOD) behavior, and fairness.
- **Inference and infrastructure:** latency, throughput, error rates, resource use, and resilience.
- **Pipeline:** CI/CD health, reproducibility, and lineage consistency.
- **Governance:** version-to-service mapping, access control, consent, and auditability in accordance with ISO/IEC 42001:2023 and AI Act Article 72 [20, 11].
- **Application and value:** user feedback, adoption, override rates, KPI impact, and cohort equity.

Each layer contributes to validating the system’s declared *value hypothesis*. Importantly, monitoring must distinguish between drift in observed reality and drift away from the system’s intended normative alignment, as improvements in predictive accuracy may still degrade fairness, legality, or public trust. A deviation in any layer can erode value, trust, or compliance, motivating corrective action through the “Decide & Adapt” loop.

### 4.7.3 Dimensions and automation

Monitoring in AIAppOps operates along three main dimensions. The *purpose dimension* concerns quality, robustness, and value alignment. The *temporal dimension* distinguishes proactive early-warning signals, reactive incident detection, and retrospective analysis for learning. The *automation dimension* defines how these observations are handled—ranging from manual reviews to fully automated responses. In practice, AIAppOps favors semi-automated setups that combine automatic detection with human judgment, balancing efficiency and accountability [8]. AIAppOps promotes semi-automated defaults, combining automated detection with human triage for safety- or value-critical systems. This balances efficiency and accountability, as recommended in [8].

#### 4.7.4 Data and architecture for monitoring

Effective monitoring depends on the systematic collection and linkage of telemetry across data, model, and inference pipelines. A robust monitoring architecture collects, links, and stores telemetry across data, model, and inference pipelines. Two principles are central: (i) *Feature and prediction logging* at inference time, including model and data version identifiers, to enable later comparison with observed outcomes; and (ii) *Lineage tracking* to guarantee that every signal can be traced to its origin and context. Such joinable records underpin reproducibility, auditability, and compliance, without reference to specific platforms, and align with best practices for structured validation and production-readiness testing in machine learning pipelines [21, 22].

#### 4.7.5 Methods and metrics

**Statistical and ML-based monitoring.** Statistical and machine-learning-based monitoring relies on metrics such as drift detection (data and concept), calibration error, uncertainty quantification, and selective prediction or abstention [23, 24, 25]. In predictive systems, these indicators preserve decision quality and safety under shifting data distributions. In generative systems, additional monitors assess groundedness, citation fidelity, toxicity and policy conformance, refusal and redaction behavior, and retrieval quality in retrieval-augmented generation (RAG). Together, these measures ensure that model outputs remain factual, appropriate, and aligned with both technical and organizational policies.

**Logical and formal monitoring.** Quantitative metrics alone cannot guarantee safety, legality, or policy compliance. Formal methods extend monitoring into the logical domain, enabling the specification and continuous verification of properties such as “the probability of violating constraint  $X$  within 5 seconds must remain below 1%”. Foundational research in runtime verification has established the theoretical and algorithmic basis for such guarantees [26, 27]. In cyber-physical and autonomous systems, temporal logic monitoring—particularly Signal Temporal Logic (STL)—has emerged as a core mechanism for specifying and measuring dynamic behaviors [28, 29, 30, 31]. These works introduced notions such as *robust satisfaction*, *partial monitoring*, and *algebraic composability*, which together make formal assurance computationally tractable in real time. Recent extensions further address uncertainty and distributional shift, combining probabilistic reasoning with runtime prediction [32].

Probabilistic Signal Temporal Logic (ProbSTL) [9] exemplifies this line of work by integrating probabilistic uncertainty with temporal constraints and providing incremental algorithms for real-time evaluation. Developed within the context of safety-aware autonomous systems [10], ProbSTL demonstrates how runtime monitoring can incorporate both stochastic predictions and confidence measures, supporting adaptive, self-assessing AI behavior. Within AIAppOps, these formalisms represent the highest maturity level of monitoring—*runtime verification under uncertainty*—where empirical observability is complemented by mathematically grounded assurance [33, 34, 35].

#### 4.7.6 Experimentation-aware releases and response

Monitoring and experimentation are coupled. New models or configurations are deployed through canary or A/B releases, where monitored metrics act as gates for promotion or rollback [2]. Detected degradations activate structured *runbooks*—predefined operational guides that specify ownership, assessment steps, and fallback or rollback actions—an established practice in Site Reliability Engineering [36, 19]. Within MLOps, such runbooks operationalize incident response and ensure consistent, accountable decision-making when monitored indicators deviate [8, 2]. Each response is logged and linked to its originating signal, ensuring traceability and learning for future iterations.

#### 4.7.7 Governance, compliance, and human oversight

Monitoring also serves as the practical mechanism for fulfilling post-market surveillance and documentation obligations. The EU AI Act requires high-risk systems to maintain a post-deployment monitoring plan that continuously assesses performance, safety, and compliance indicators [11]. International standards such as ISO/IEC 23894:2023 and 42001:2023 reinforce this principle by mandating continuous improvement cycles. In AIAppOps, dashboards are designed not only for observability but for *interpretability*—presenting uncertainty visualizations and explanatory cues that allow human operators to calibrate trust and intervene when necessary. In this way, AIAppOps generalizes assurance principles from system-level runtime verification [10] to organizational oversight, ensuring that compliance and human judgment remain anchored in verifiable evidence.

This approach is consistent with broader AI assurance frameworks [33, 37, 38], linking continuous monitoring with calibrated human trust and accountable oversight.

#### 4.7.8 Maturity roadmap

Monitoring capability evolves with organizational maturity:

- **Levels 0–1:** ad-hoc checks; manual dashboards for latency and availability.
- **Level 2:** centralized metrics; data-drift detection and feature/prediction logging.
- **Level 3:** automated alerting, calibration and OOD monitors, A/B testing, and retrain triggers.
- **Level 4:** unified dashboards integrating business KPIs, fairness and compliance indicators, and formal runtime verification.

Advancing along these maturity levels should follow demonstrated operational need and organizational competence. Premature automation risks obscuring accountability and undermining trust—two of the very outcomes monitoring is designed to preserve.

The signals and decisions established through monitoring directly feed the *Observe* and *Decide & Adapt* activities described in Section 5, ensuring that every observed deviation—technical or organizational—is tied to measurable value outcomes.

### Monitoring value connection

Monitoring transforms observability into value control. Data and model signals (drift, calibration, OOD) anticipate value loss before KPIs move; inference SLOs safeguard timeliness and user experience; application metrics and user feedback close the loop on realized outcomes. Feature- and prediction-level lineage enables precise attribution when value drifts, focusing improvements where they yield the highest return [2, 8].

## 4.8 Assess, Analyze, Improve

Continuous assessment within AIAppOps extends beyond technical metrics to a full socio-technical learning process that links operational signals to organizational improvement. Every observed deviation—be it technical, procedural, or ethical—initiates a structured cycle of assessment, analysis, and improvement.

**Assessment** refers to the continuous evaluation of monitored signals across the data, model, inference, and application layers. It determines whether observed deviations affect the declared value hypothesis or trustworthiness requirements. Examples include drift in data distributions, loss of calibration, or shifts in user behavior.

**Analysis** focuses on understanding root causes and their implications. Technical causes (e.g., concept drift, data imbalance, or model overfitting) are linked to organizational and process factors (e.g., missing validation steps, insufficient human oversight, or inadequate documentation). This diagnostic step is crucial for preventing repeated errors and for turning incidents into learning opportunities.

**Improvement** transforms insights into structured change. This may include retraining models, updating labeling practices, improving monitoring thresholds, refining user interfaces, or adjusting governance rules. Each improvement is logged, versioned, and validated, ensuring reproducibility and accountability.

Over time, this cyclical process increases both *AI maturity* and *organizational resilience*. It evolves from reactive error correction to proactive optimization and anticipatory governance. By explicitly linking technical evidence to human decision-making and value outcomes, AIAppOps ensures that improvement efforts are data-driven, ethical, and aligned with organizational goals.

## 4.9 Trustworthy AI and Compliance

Trustworthy AI in the European context is built upon three mutually reinforcing pillars: **lawfulness**, **ethical alignment**, and **technical robustness**. These pillars, defined in the European

Commission’s *Ethics Guidelines for Trustworthy AI* [17] and reinforced by the *EU Artificial Intelligence Act* [11], establish a foundation for responsible AI development and deployment. AIAppOps operationalizes these principles throughout the AI lifecycle by embedding them in its processes for data governance, modeling, monitoring, and value-driven improvement.

Trustworthy AI is not a static certification but a continuous state achieved through deliberate monitoring, documentation, and adaptation. In AIAppOps, these requirements are mapped to operational mechanisms as follows:

### **1. Human Agency and Oversight**

AIAppOps establishes *human-in-the-loop (HITL)* and *human-on-the-loop (HOTL)* modes to preserve situational awareness and control, in line with EU AI Act Article 14. Decision deferral mechanisms and calibrated uncertainty visualizations empower operators to interpret and override outputs. Dashboards fuse reliability, explainability, and confidence signals to support informed human judgment.

### **2. Technical Robustness and Safety**

Reliability under uncertainty is assured through *probabilistic runtime verification (Prob-STL)*, calibration and out-of-distribution monitoring, adversarial robustness testing, and reproducible builds. Fail-safe fallback and recovery policies sustain traceability and operational continuity during abnormal behavior.

### **3. Privacy and Data Governance**

Data pipelines integrate GDPR-aligned lineage tracking, retention, and consent management (AI Act Article 10). Automated minimization, pseudonymization, and masking are enforced within CI/CD flows, while access logging and accountability policies ensure lawful, transparent data use.

### **4. Transparency**

Standardized documentation—*Data Sheets*, *Model Cards*, and *Decision Logs*—make datasets, models, and decisions auditable (AI Act Article 13). Versioned lineage provides traceable insight into system assumptions, capabilities, and limitations across releases.

### **5. Diversity, Non-discrimination, and Fairness**

Bias detection, mitigation, and accessibility testing are embedded across data curation, model evaluation, and monitoring. Their meaning depends on the declared data alignment intent: whether the system describes historical reality or enforces normative decision principles. Diverse datasets and user cohort testing ensure equitable performance and fair treatment across demographics and contexts.

### **6. Societal and Environmental Well-being**

AIAppOps tracks environmental impact (energy per inference, training footprint) and relates system performance to societal indicators. This embeds sustainability and democratic accountability within operational metrics, ensuring AI contributes positively to long-term societal resilience.

### **7. Accountability**

Clear ownership of artifacts, roles, and lifecycle stages ensures responsibility. Audit logs, compliance dashboards, and ISO/IEC 42001:2023 alignment provide evidence of continuous stewardship and regulatory conformity throughout operation.

By aligning process design and monitoring infrastructure with these seven requirements, AIAppOps transforms abstract ethical principles into verifiable operational practices. Compliance is not treated as an external audit phase, but as an *embedded operational property*—maintained through continuous feedback, documented evidence, and multi-role accountability.

At higher organizational maturity, trustworthy AI becomes inseparable from operational excellence: systems are not only safe and lawful, but also demonstrably aligned with human values, societal benefit, and long-term sustainability.

## Lawful, Ethical, and Robust AI — The Foundations of Trustworthy AI

- **Lawful AI:** Complies with the EU AI Act, GDPR, and ISO/IEC 42001. Ensures documentation, human oversight, and accountability.
- **Ethical AI:** Promotes fairness, transparency, and human agency, avoiding bias and misuse.
- **Robust AI:** Maintains safety, resilience, and explainability under uncertainty through probabilistic and formal monitoring.

Together, these principles define the socio-technical baseline that AIAppOps operationalizes through its continuous lifecycle.

## 5 Value-driven Operations

This section defines the operational loop that links technical observability to business outcomes. It consists of four recurring activities—*Hypothesize, Instrument, Observe, Decide & Adapt*—executed at increasing scope as systems mature (use case → product → portfolio).

### 5.1 Hypothesize

Teams articulate a value hypothesis that specifies (i) intended outcomes and beneficiaries, (ii) measurable proxies and decision thresholds, and (iii) constraints and guardrails.

- **Outcomes:** financial (cost/revenue), operational (throughput, latency, quality), risk/compliance (incidents avoided, audit timeliness), and public value (access, equity, satisfaction).
- **Proxies and thresholds:** early-stage technical metrics that predict target outcomes (e.g., calibration error → triage safety; groundedness → casework quality) with pre-agreed thresholds that gate POC→MVP→rollout.
- **Guardrails:** safety, privacy, policy, and legal requirements that must hold at each stage (e.g., data minimization, audit trails, transparency obligations).

### 5.2 Instrument

Before building, teams ensure that value is *measurable* and *traceable*.

- **Telemetry plan:** event schemas for task outcomes, human overrides/deferrals, explanation/justification capture, and user feedback; cohort and feature flags for experiments.
- **Evaluation harness:** offline test suites (predictive: cost-sensitive, calibration, drift; generative: task/eval sets, groundedness, refusal/policy tests) aligned to the hypothesis.
- **Lineage and auditability:** versioning for data, code, models, prompts/policies, and configurations; reproducible builds; immutable logs that link predictions/generations to inputs and model versions.
- **Privacy and compliance hooks:** access controls, PII handling, consent/accountability records, and documentation artifacts integrated into CI/CD.

### 5.3 Observe

Run staged experiments and operations with decision-quality visibility.

- **Staged delivery:** POC in a sandbox or shadow mode; MVP to limited cohorts; progressive rollout with canarying and A/B.
- **Dashboards:** joint views that bind technical signals (latency, error, drift, hallucination/failure modes) to business KPIs (e.g., handle time, deflection, risk events).
- **Degradation and drift:** SLO/SLA alerts tied to value impact; statistical and semantic drift detectors that trigger upstream investigation.

## 5.4 Decide & Adapt

When signals deviate (positively or negatively), changes are targeted where they have the highest value leverage.

- **Data:** re-sample cohorts, re-label, close coverage gaps, update retention/freshness SLAs.
- **Model:** adjust loss/thresholds, retrain or fine-tune, introduce abstention or selective routing, expand evaluation suites.
- **Inference:** change routing policies, enable fallbacks (e.g., retrieval-only answers), adjust autoscaling and caching to hit value-critical SLOs.
- **Application:** modify UX and integration points, change human-in-the-loop placement, evolve policy prompts and explanations.

Decisions and their rationale are recorded with links to metrics, artifacts, and approvals to maintain traceability.

## 5.5 Operating cadence and ownership

Value is a product property with clear ownership.

- **Roles:** a business owner (accountable for outcomes) and a technical owner (accountable for service and model behavior) jointly steward the value hypothesis.
- **Cadence:** weekly operational reviews of SLOs and safety signals; monthly value reviews of KPI movement and cohort equity; quarterly portfolio reprioritization based on realized value and risk.
- **Change management:** pre-commit rollback criteria; maintain model/prompt/policy change logs; ensure documentation (model cards, data sheets, decision records) is updated as part of CI/CD.

This cadence closes the loop between observed impact and technical change, ensuring that deployed AI systems remain safe, effective, and valuable over time. Where the monitoring layer provides probabilistic and formal assurance of technical behavior, the value-driven loop ensures that these assurances translate into measurable organizational and societal outcomes.

# 6 Conclusion and Outlook

AI Application Operations (AIAppOps) establishes a socio-technical framework that extends traditional MLOps to encompass the full lifecycle of AI systems—from initial value hypothesis through data, model, inference, and application, to post-deployment monitoring and improvement. Its defining feature is the continuous feedback between these processes, guided by a value-driven meta-process that ensures every technical activity contributes to measurable and sustainable outcomes.

By embedding observability, uncertainty awareness, and human oversight into all lifecycle stages, AIAppOps bridges the gap between engineering practice and trustworthy AI principles. It aligns with the European Union’s vision of *lawful, ethical, and robust* AI as described in the *Ethics Guidelines for Trustworthy AI* and the *EU AI Act*, operationalizing their requirements through concrete, monitorable processes.

The framework is intentionally modular and adaptable: it can be implemented incrementally according to organizational maturity, domain criticality, and regulatory obligations. Organizations at early maturity levels can begin with data readiness and monitoring foundations, while mature AI programs can integrate formal verification, probabilistic assurance, and full governance alignment.

**Looking forward**, several directions emerge for advancing both research and practice:

- *Standardization:* Convergence between AIAppOps and ISO/IEC 42001, 23894, and future AI assurance standards can formalize the operational backbone of trustworthy AI.
- *Automation:* Integrating runtime reasoning and formal verification (e.g., ProbSTL) with adaptive retraining loops can yield self-assessing AI systems that maintain safety and alignment autonomously.

- *Open ecosystems*: Shared reference implementations, open-source observability stacks, and benchmarking of AI monitoring architectures will accelerate cross-sector adoption.
- *Human oversight at scale*: Designing interfaces that communicate uncertainty, provenance, and rationale effectively to human operators remains a frontier for socio-technical AI design.

AIAppOps thus provides not only a practical framework for organizations today but also a foundation for tomorrow’s AI governance infrastructures—where value, trust, and accountability evolve together through evidence-based operations.

`sectionAcknowledgements` We would like to especially thank Mattias Jonhede for feedback on the paper and discussions, Daniel Jakobsson for valuable discussions and inspiration about machine learning processes at the Swedish Transport Administration. This work was financed by VINNOVA and the organization participating in the Data-Driven Organizations (DDO) project. The organizations that participated in DDO were Aixia, Hewlett Packard Enterprise, Hopsworks, IBM, Linköping University, NetApp, Predli, Proact, RISE, RedHat, Region Halland, Sahlgrenska University Hospital, Statistics Sweden (Statistiska Centralbyrån), The Swedish Tax Agency (Skatteverket), Stormgrid, The Swedish Transport Administration (Trafikverket), Volvo Parts, Region Västra Götaland, Santa Anna, and AI Sweden.

## Glossary of Key Terms and Concepts

**Abstention:** The ability of a model to decline making a prediction when confidence is low.

**Example:** A medical diagnosis model might abstain when uncertainty exceeds a threshold, triggering human review.

**Why it matters:** Abstention improves safety and trustworthiness, especially in high-stakes domains.

**Bias and Fairness:** Bias refers to systematic errors that unfairly disadvantage certain groups; fairness refers to mitigation strategies and equitable outcomes.

**Example:** A hiring model that under-selects women due to biased training data exhibits gender bias.

**Why it matters:** Addressing bias is critical for legal compliance, ethical responsibility, and public trust.

**Calibration:** The alignment between a model's predicted probabilities and actual outcomes.

**Example:** A calibrated model predicting 0.8 probability of churn will see roughly 80% of those cases churn in reality.

**Why it matters:** Calibration ensures that confidence scores are meaningful, supporting risk-sensitive decisions and uncertainty estimation.

**Canary Deployment:** A deployment strategy where a new version of a model or service is rolled out to a small subset of users or requests before full release.

**Example:** Only 5% of traffic is routed to a new model initially, allowing engineers to monitor its behavior before scaling up.

**Why it matters:** Canary deployments reduce risk by detecting performance or compliance issues early in the release process.

**CI/CD (Continuous Integration / Continuous Deployment):** A set of software engineering practices that automate the building, testing, and deployment of code and models into production.

**Example:** A CI/CD pipeline might automatically retrain a model when new data arrives, test it against validation criteria, and deploy it to a staging environment.

**Why it matters:** CI/CD accelerates iteration, reduces human error, and ensures that models and services can evolve continuously without sacrificing reliability.

**Concept Drift:** A change in the relationship between input features and the target variable over time.

**Example:** A spam classifier may degrade as spammers adopt new tactics that change the underlying concept of "spam."

**Why it matters:** Concept drift requires model retraining or adaptation to maintain predictive performance.

**Containerization:** Packaging software and its dependencies into lightweight, portable containers to ensure consistent execution across environments.

**Example:** A model inference service packaged in a Docker container behaves identically in development, testing, and production environments.

**Why it matters:** Containerization simplifies deployment, scaling, and reproducibility, which are essential for reliable AI operations.

**Data Catalog:** A searchable inventory of datasets and metadata, often enriched with lineage, quality metrics, and ownership information.

**Example:** A catalog entry might describe a customer dataset, its schema, last update, and associated access policies.

**Why it matters:** Data catalogs make data discoverable and governable, accelerating development and reducing duplication.

**Data Drift:** A change in the statistical properties of input data over time, which can degrade model performance.

**Example:** A credit scoring model trained on pre-pandemic data may become inaccurate as customer behavior shifts post-pandemic.

**Why it matters:** Detecting and responding to data drift is crucial for maintaining model accuracy and reliability in production.

**Data Governance:** The set of policies, roles, and processes that ensure data is used responsibly, securely, and in compliance with regulations.

**Example:** A governance framework may define data access controls, retention policies, and consent management procedures.

**Why it matters:** Strong governance underpins trust, legal compliance, and operational reliability.

**Data Sheet:** A standardized documentation format describing a dataset's origin, composition, collection process, and potential risks.

**Example:** A medical dataset's data sheet might document patient consent procedures and known sampling biases.

**Why it matters:** Data sheets help assess suitability, legal compliance, and ethical risks associated with data use.

**Explainability:** Techniques and tools that make model decisions understandable to humans.

**Example:** SHAP values might explain which features, for a specific learned model, most influenced a loan approval decision.

**Why it matters:** Explainability supports trust, debugging, accountability, and compliance with regulations like the AI Act.

**Feature Store:** A centralized repository for storing, managing, and serving machine learning features consistently across training and inference.

**Example:** A feature store might provide preprocessed customer features to both a churn prediction model and a recommendation engine.

**Why it matters:** Feature stores improve consistency, reuse, and governance of data pipelines, accelerating development and reducing errors.

**Feedback Loop:** A mechanism by which model predictions and user interactions are fed back into training or evaluation processes.

**Example:** User corrections to an AI assistant's answers can be used to improve future model performance.

**Why it matters:** Feedback loops enable continuous improvement and adaptation to real-world conditions.

**Hallucination:** The generation of plausible-sounding but incorrect or fabricated information by a generative model.

**Example:** An LLM might invent a nonexistent academic reference in its response.

**Why it matters:** Hallucinations undermine trust and can have legal, safety, or reputational consequences if unmitigated.

**Human-in-the-loop (HITL):** A system design where humans review, correct, or override AI decisions.

**Example:** A content moderation system may flag posts for human review instead of automatically removing them.

**Why it matters:** HITL improves safety, reduces errors, and ensures compliance in sensitive applications.

**KPI (Key Performance Indicator):** A measurable metric that indicates how well a system or process is achieving its objectives.

**Example:** A fraud detection system’s KPI might be the percentage of fraudulent transactions caught without false positives.

**Why it matters:** KPIs connect technical performance to business outcomes, enabling meaningful evaluation of AI’s value.

**Lineage:** The traceable record of how data, models, and artifacts were created, transformed, and used over time.

**Example:** Lineage tracking can show which dataset version and preprocessing pipeline produced a given model version.

**Why it matters:** Lineage supports reproducibility, auditing, and regulatory compliance.

**LLMOps (Large Language Model Operations):** A specialization of MLOps focused on developing, deploying, and maintaining large language models and generative AI systems.

**Example:** LLMOps workflows might manage prompt templates, fine-tuned checkpoints, and hallucination monitoring.

**Why it matters:** LLMOps addresses the unique challenges of generative AI, from prompt versioning to content safety.

**MLOps (Machine Learning Operations):** The set of practices, tools, and cultural approaches that unify machine learning system development and operations.

**Example:** A team using MLOps might automate data ingestion, model training, deployment, and monitoring in a continuous workflow.

**Why it matters:** MLOps is foundational for scaling AI from prototypes to reliable, maintainable production systems.

**Model Card:** A standardized documentation format that describes a model’s purpose, data, performance, limitations, and ethical considerations.

**Example:** A vision model’s model card might include training data sources, known biases, accuracy by demographic, and appropriate use cases.

**Why it matters:** Model cards improve transparency, governance, and stakeholder communication.

**Observability:** The ability to understand a system’s internal state based on external outputs such as logs, metrics, and traces.

**Example:** Observability tools might track inference latency, model confidence scores, and data drift metrics over time.

**Why it matters:** Observability enables proactive monitoring, rapid debugging, and informed decision-making in production AI systems.

**Pipeline:** A sequence of automated steps for data processing, training, validation, deployment, and monitoring.

**Example:** A training pipeline might ingest raw data, preprocess it, train a model, evaluate performance, and register the model.

**Why it matters:** Pipelines enforce reproducibility, scalability, and automation across the AI lifecycle.

**POC (Proof of Concept):** A small-scale experiment or prototype used to test feasibility before committing significant resources.

**Example:** A team might build a POC chatbot using open-source models before investing in a production deployment.

**Why it matters:** POCs reduce risk and inform decision-making in early project stages.

**Post-deployment Monitoring:** The continuous observation of model behavior, inputs, outputs, and outcomes after deployment.

**Example:** Monitoring might detect performance degradation due to data drift or unexpected user behavior.

**Why it matters:** Post-deployment monitoring is essential for sustaining performance, safety, and compliance over time.

**Prompt Engineering:** The practice of crafting and refining prompts to guide the behavior of large language models (LLMs).

**Example:** A customer support chatbot might respond more accurately when prompted with structured instructions and examples.

**Why it matters:** Effective prompt engineering is essential for controlling generative AI systems and aligning them with desired outcomes.

**RAG (Retrieval-Augmented Generation):** A technique that combines a generative model with an external knowledge retrieval step to improve accuracy and grounding.

**Example:** A legal assistant LLM may retrieve relevant case law from a database before generating an answer.

**Why it matters:** RAG improves factual reliability and reduces hallucinations in generative AI applications.

**Retraining Cycle:** The periodic process of retraining models to incorporate new data or address drift.

**Example:** A recommendation system might retrain weekly using the latest user interaction data.

**Why it matters:** Regular retraining ensures models remain accurate, relevant, and aligned with changing realities.

**Rollback:** Reverting a deployed model or service to a previous version after e.g. detecting performance or compliance issues.

**Example:** A rollback might occur if a new model version causes error rates to spike in production.

**Why it matters:** Rollbacks are a safety mechanism that protect business continuity and reliability.

**Shadow Deployment:** A deployment strategy where a new model runs in parallel with the current production model but does not affect outputs.

**Example:** A bank might run a new fraud detection model in shadow mode to compare its decisions against the production model without impacting customers.

**Why it matters:** Shadow deployments allow safe real-world evaluation of new models before switching them into production.

**SLA (Service Level Agreement):** A contractual or internal agreement specifying measurable performance targets for a service, such as availability, latency, or throughput.

**Example:** An inference API might have an SLA guaranteeing 99.9% uptime and response times under 200 ms.

**Why it matters:** SLAs define expectations and accountability between teams or organizations, ensuring that deployed AI services meet operational requirements and business needs.

**SLO (Service Level Objective):** A specific, measurable performance target used internally to track service quality and adherence to SLAs.

**Example:** A model-serving service may define an SLO of 99.95% uptime, which supports achieving the 99.9% SLA promised to customers.

**Why it matters:** SLOs provide engineering teams with concrete, actionable goals that align system performance with business commitments.

**Synthetic Data:** Artificially generated data used to train or test models without exposing real sensitive data.

**Example:** A healthcare company might use synthetic patient records to test models while preserving privacy.

**Why it matters:** Synthetic data expands training options, mitigates privacy risks, and supports regulatory compliance.

**Value Drift:** The divergence between an AI system’s outputs and the evolving sources of business or societal value it was meant to serve.

**Example:** A recommendation system optimized for engagement may harm user trust over time, reducing actual value.

**Why it matters:** Monitoring for value drift ensures AI remains aligned with strategic goals as conditions change.

**Value Hypothesis:** An explicit assumption about how an AI system will generate value for the organization or users.

**Example:** A predictive maintenance model is expected to reduce unplanned downtime by 30%.

**Why it matters:** A clear value hypothesis guides design decisions and provides a basis for measuring success.

## References

- [1] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, *CRISP-DM 1.0: Step-by-Step Data Mining Guide*, CRISP-DM Consortium (SPSS, NCR, DaimlerChrysler, OHRA), 2000. [Online]. Available: <https://the-modeling-agency.com/crisp-dm.pdf>
- [2] K. Salama, J. Kazmierczak, and D. Schut, “Practitioners guide to MLOps: A framework for continuous delivery and automation of machine learning,” Google Cloud White Paper, 2021. [Online]. Available: <https://cloud.google.com/resources/mlops-whitepaper>
- [3] D. Kreuzberger, N. Köhl, and S. Hirschl, “Machine learning operations (mlops): Overview, definition, and architecture,” *IEEE Access*, vol. 11, pp. 31 866–31 879, 2023.
- [4] N. D. Lawrence, “Data readiness levels,” *arXiv preprint arXiv:1705.02245*, 2017. [Online]. Available: <https://arxiv.org/abs/1705.02245>
- [5] S. Afzal, C. Rajmohan, M. Kesarwani, S. Mehta, and H. Patel, “Data readiness report,” in *2021 IEEE international conference on smart data services (SMDS)*. IEEE, 2021, pp. 42–51. [Online]. Available: <https://arxiv.org/pdf/2010.07213>
- [6] M. Tiger *et al.*, “Exploratory visual analysis for increasing data readiness in ai projects,” Manuscript/whitepaper, 2024, preprint/unpublished; provide venue/DOI om tillgängligt.
- [7] J. Jakubik, M. Vössing, N. Köhl, J. Walk, and G. Satzger, “Data-centric artificial intelligence,” *Business & Information Systems Engineering*, 2024.
- [8] H. Naveed, S. Barnett, C. Arora, J. Grundy, H. Khalajzadeh, and O. Haggag, “Monitoring machine learning systems: A multivocal literature review,” *arXiv preprint arXiv:2509.14294*, 2025. [Online]. Available: <https://arxiv.org/abs/2509.14294>
- [9] M. Tiger and F. Heintz, “Incremental reasoning in probabilistic signal temporal logic,” *International Journal of Approximate Reasoning*, vol. 119, pp. 325–352, 2020.
- [10] M. Tiger, *Safety-Aware Autonomous Systems: Preparing Robots for Life in the Real World*, ser. Frontiers in Artificial Intelligence and Applications. IOS Press, 2023, vol. 373.
- [11] European Parliament and Council, “Artificial intelligence act,” 2024, regulation (EU) 2024/1689, 13 June 2024. [Online]. Available: <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>
- [12] O. Spjuth, J. Frid, and A. Hellander, “The machine learning life cycle and the cloud: Implications for drug discovery,” *Expert Opinion on Drug Discovery*, vol. 16, no. 9, pp. 1071–1079, 2021.

- [13] S. Shankar, R. Garcia, J. M. Hellerstein, and A. G. Parameswaran, ““we have no idea how models will behave in production until production”: How engineers operationalize machine learning,” *Proceedings of the ACM on Human-Computer Interaction*, vol. 8, no. CSCW1, 2024, article 206. [Online]. Available: <https://arxiv.org/abs/2403.16795>
- [14] M. M. John, H. H. Olsson, and J. Bosch, “An empirical guide to mlops adoption: Framework, maturity model and taxonomy,” *Information and Software Technology*, vol. 183, p. 107725, 2025.
- [15] —, “Towards MLOps: A framework and maturity model,” in *47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2021, pp. 1–8.
- [16] Microsoft, “Machine learning operations maturity model,” Microsoft Learn, 2025, accessed 2025-10-13. [Online]. Available: <https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/mlops-maturity-model>
- [17] High-Level Expert Group on Artificial Intelligence, “Ethics guidelines for trustworthy ai,” <https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai>, 2019, european Commission, Brussels.
- [18] M. Feurer, A. Klein, K. Eggenberger, J. T. Springenberg, M. Blum, and F. Hutter, “Efficient and robust automated machine learning,” in *Advances in Neural Information Processing Systems 28 (NeurIPS 2015)*, 2015. [Online]. Available: <https://proceedings.neurips.cc/paper/2015/hash/11d0e6287202fcd83f79975ec59a3a6-Abstract.html>
- [19] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, “Devops,” *IEEE Software*, vol. 33, no. 3, pp. 94–100, 2016.
- [20] *ISO/IEC 42001:2023 — Artificial Intelligence Management System*, International Organization for Standardization Std., 2023. [Online]. Available: <https://www.iso.org/standard/81230.html>
- [21] E. Breck, S. Cai, E. Nielsen, M. Salib, and D. Sculley, “The ML test score: A rubric for ML production readiness and technical debt reduction,” in *2017 IEEE International Conference on Big Data (Big Data)*, 2017, pp. 1123–1132.
- [22] A. Odena and et al., “Tensorflow data validation: Data analysis and validation in ml pipelines,” in *SysML Workshop*, 2019.
- [23] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 1321–1330. [Online]. Available: <https://proceedings.mlr.press/v70/guo17a.html>
- [24] D. Hendrycks and K. Gimpel, “A baseline for detecting misclassified and out-of-distribution examples in neural networks,” *ICLR*, 2017.
- [25] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” in *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, 2017. [Online]. Available: <https://papers.nips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html>
- [26] K. Havelund and G. Roşu, “Runtime verification: Past, present, and future,” *Software Tools for Technology Transfer*, vol. 20, no. 2, pp. 237–249, 2018.
- [27] Y. Falcone, C. Colombo *et al.*, “A tutorial on runtime verification,” *Communications of the ACM*, vol. 62, no. 2, pp. 92–102, 2019.
- [28] A. Donzé, T. Ferrère, and O. Maler, “Efficient robust monitoring for STL,” in *Computer Aided Verification (CAV)*. Springer, 2013, pp. 264–279.
- [29] J. Deshmukh, A. Donzé, S. A. Seshia, and X. Jin, “Robust online monitoring of signal temporal logic,” in *Runtime Verification (RV)*. Springer, 2015, pp. 55–70.
- [30] S. Mitsch and A. Platzer, “Verified runtime validation for partially observable hybrid systems,” *Formal Methods in System Design*, vol. 52, pp. 1–35, 2018.

- [31] S. Jaksic, E. Bartocci, R. Grosu, and D. Nickovic, “Algebraic runtime verification,” *International Journal on Software Tools for Technology Transfer*, vol. 20, no. 2, pp. 215–232, 2018.
- [32] Y. Zhao, C. Arzate Cruz, and S. A. Seshia, “Robust conformal prediction for STL runtime verification under distribution shift,” *arXiv preprint arXiv:2311.09482*, 2023. [Online]. Available: <https://arxiv.org/abs/2311.09482>
- [33] K. R. Varshney and H. Alemzadeh, “On the safety of machine learning: Cyber-physical systems, decision sciences, and data products,” *Big Data*, vol. 5, no. 3, pp. 246–255, 2017.
- [34] J. Zhang *et al.*, “Formal methods for AI systems,” *Nature Machine Intelligence*, vol. 2, no. 8, pp. 483–484, 2020.
- [35] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, “Concrete problems in AI safety,” *arXiv preprint arXiv:1606.06565*, 2016. [Online]. Available: <https://arxiv.org/abs/1606.06565>
- [36] B. Beyer, C. Jones, J. Petoff, and N. R. Murphy, *Site Reliability Engineering: How Google Runs Production Systems*. O’Reilly Media, 2016.
- [37] F. Doshi-Velez and B. Kim, “Towards a rigorous science of interpretable machine learning,” *arXiv preprint arXiv:1702.08608*, 2017. [Online]. Available: <https://arxiv.org/abs/1702.08608>
- [38] A. Bussone, S. Stumpf, and D. O’Sullivan, “The role of explanations on trust and reliance in clinical decision support systems,” *BMC Medical Informatics and Decision Making*, vol. 15, no. 1, p. 40, 2015.