Evaluation for Artificial Capable Intelligence (ACI) Architecture Reviews (Simple Template)

Purpose

This template provides a basic structured framework to evaluate an **Artificial Capable Intelligence (ACI)** system's architecture through the phases of **Scoping**, **Evaluation**, **Reporting**, and **Review**. It is aligned with core ACI principles and key requirements – *data integrity, seamless integration, responsible AI, and capability enablement* – to ensure the design delivers practical value responsibly. This basic template can be used as part of the methodology released by Fede Nolasco in May 2025 called "*A Methodology for Evaluating Artificial Capable Intelligence (ACI) Architectures (Platform-Agnostic),"* (1). Each section below includes checklist questions (with **Yes/No/Partial** scoring and comments) and helper descriptions to guide a comprehensive assessment.

Evaluation Metadata

Evaluator Name:	**Organization:**
Date:	**Approver:**

Fill in the above fields for each ACI evaluation. The Approver is the authority who will review and sign off on the evaluation outcome.

Scoping Phase

In the Scoping phase, clearly define **what** will be evaluated and **why**. Establish the system boundaries, objectives, and context before proceeding. Ensure all necessary information and stakeholders are prepared.

Scoping Criteria ***(Confirm preparatory steps before evaluation)***	**Yes/No/Partial**	**Comments**
Scope & Objectives Defined – Have the ACI system (or component) under review and the specific evaluation objectives been clearly identified? *Helps ensure everyone understands the system boundaries, intended use, and goals of the evaluation.*		

Scoping Criteria ***(Confirm preparatory steps before evaluation)***	**Yes/No/Partial**	**Comments**
Artifacts Collected – Are all relevant architecture artifacts gathered (architecture diagrams, data flow diagrams, specifications, model documentation, policies, prior test results, risk assessments)? *These materials provide evidence and context needed for the assessment.*		
Stakeholders Identified – Has a diverse set of key stakeholders been assembled or consulted (e.g. architects, developers, data scientists, business owners, security/privacy officers)? *Engaging stakeholders ensures a well-rounded perspective and access to needed expertise.*		

Evaluation Phase

Instructions*:* For each criterion in the evaluation dimensions below, indicate **Yes** (criterion fully met), **No** (not met), or **Partial** (partially met or in progress). Provide **comments** explaining evidence, specifics, or plans. The Evaluation phase is organized into four key architectural dimensions derived from the ACI methodology:

- **Data Integrity & Governance Architecture** – Ensuring data quality, security, and governance throughout the AI lifecycle.
- **Seamless Integration Architecture** – Ensuring the AI system integrates well with other systems and processes.
- **Responsible AI Architecture** – Ensuring ethical, secure, fair, and compliant AI operation by design.
- **Capability Enablement & Value Realization** – Ensuring the architecture delivers the intended functionality, performance, and business value.

Each dimension is introduced with a brief description, followed by specific checklist criteria.

Data Integrity and Governance Architecture

This dimension assesses how the architecture ensures high-quality data that is handled securely and ethically, flowing efficiently through its lifecycle. It addresses ACI principles like "Data as Fuel" – treating data as a critical asset – and incorporates "Responsibility by Design" in data management.

Data Quality Management – *Are there measures to maintain and improve data quality?*

Criterion & Description	Yes/No/Partial	Comments
Mechanisms for Data Quality – Does the architecture implement robust data sourcing, validation rules, cleansing processes, continuous quality monitoring, and data lineage tracking to ensure input data is accurate and reliable? *These mechanisms should cover all data entering the system (including both structured and unstructured data) and provide ongoing validation and cleaning to prevent quality issues.*		
Handling of Diverse Data Types – Does the architecture support quality management for **both** structured and unstructured data? *Ensure that data integrity checks and preprocessing are in place for all relevant data formats (text, images, etc.), as unstructured data is often crucial for AI applications.*		
ML Pipeline Integration – Are data preparation and feature engineering stages well-integrated into the architecture's machine learning pipeline? *The design should maintain data integrity through transformations (e.g., no loss or corruption of data) and support versioning of datasets/features for reproducibility.*		

Criterion & Description	Yes/No/Partial	Comments
Automated Quality Checks – Does the architecture support automated data quality checks and anomaly detection? *Automation (e.g., automated validation scripts, anomaly detection tools, metadata tagging) is essential to manage data quality at scale.*		

Data Governance Enforcement – *How does the architecture enforce data governance policies?*

Criterion & Description	Yes/No/Partial	Comments
Policies & Access Controls – Are data access controls and usage policies enforced within the architecture (e.g., role-based access control, data privacy rules, retention schedules)? *Check for integration with security policies (e.g., GDPR/CCPA compliance) and whether data is accessible only by authorized roles (see **RBAC** in Definitions). Proper controls should prevent unauthorized access or misuse.*		
Stewardship & Framework Integration – Does the architecture support data stewardship roles and integrate with enterprise data governance frameworks or tools? *This could include support for data cataloging, metadata management, or alignment with standards (e.g., DAMA-DMBOK). Clear ownership and governance processes should be facilitated by the design.*		
Traceability & Audit – Are there mechanisms to trace data lineage and provide audit trails of data usage and changes? *Effective **traceability** means any data item can be tracked from ingestion to usage in models, with logs for who accessed or changed data. This is crucial for compliance and accountability.*		

Data Accessibility & Flow – *Does the architecture enable efficient, secure data flow throughout the system?*

Criterion & Description	Yes/No/Partial	Comments
Data Pipelines – Are data pipelines designed for efficient, secure, and scalable movement of data into and out of AI models? *Evaluate how data is ingested, transformed, and fed to models, and how results are delivered back. Pipelines should be reliable and handle the needed data volume with proper error handling and security (encryption in transit, etc.).*		
Lifecycle Support – Does the architecture provide appropriate data storage solutions and version control throughout the data lifecycle? *Check if there are provisions for data ingestion, suitable storage (databases, data lakes) considering performance and cost, versioning of datasets for ML reproducibility, and processing capabilities for large data sets.*		
Real-Time Data Handling – If real-time data is required, does the architecture support streaming or real-time processing? *For use cases needing low-latency data (e.g., live sensor feeds), verify integration of streaming technologies or message queues and how governance is applied to streaming data.*		
Prevention of Data Silos – Does the design avoid data silos and promote a “single source of truth” where appropriate? *Assess whether different components share data consistently or if data gets isolated. The architecture should encourage unified data views or interoperable data stores to prevent fragmentation.*		

Seamless Integration Architecture

This dimension evaluates how well the ACI system's components integrate with each other and with the broader IT environment. The architecture should treat integration with existing systems, data sources, and workflows as a primary concern ("Integration First" principle), enabling end-to-end business process enhancement instead of isolated AI functionality.

API Strategy – *Are interfaces and APIs designed to enable modular, secure integration?*

Criterion & Description	Yes/No/Partial	Comments
API Design & Management – Are the system's APIs well-designed, documented, and managed? *Look for clear, stable API contracts for services or models, with versioning and discovery mechanisms. Good API management (possibly via gateways or management platforms) indicates the architecture is integration-ready.*		
Modularity & Connectivity – Does the API design promote modularity of components? *The architecture should allow components to be updated or replaced independently via APIs. In other words, APIs act as the "connective tissue" between components, enabling loose coupling and flexibility.*		
API Security – Are robust security measures in place for APIs (authentication, authorization, rate limiting, input validation, etc.)? *APIs are often a key attack surface. Confirm the use of API keys/OAuth, role-based access, throttling to prevent abuse, and validation to mitigate injections or other attacks.*		

System Interoperability – *How easily does the system connect with external systems and legacy components?*

Criterion & Description	Yes/No/Partial	Comments
Integration Mechanisms – Does the architecture include appropriate mechanisms (middleware, message buses, adapters) to connect with existing enterprise systems (e.g., CRM, ERP, databases, legacy apps)? *Check if integration patterns like messaging queues, event streams, or API adapters are used to seamlessly bridge the ACI system with other systems.*		
Standards & Protocols – Does the system use standard interfaces/protocols (REST, JSON, gRPC, etc.) to facilitate interoperability? *Relying on industry-standard communication formats and protocols makes plugging into the existing environment easier. Evaluate if any proprietary or unusual interfaces could hinder integration.*		
Hybrid Environment Support – Can the architecture support integrations across cloud and on-premise environments if required? *If the AI components or data sources span cloud and local data centers, confirm the design can handle network considerations, security (VPNs, etc.), and latency for a hybrid integration.*		

Workflow Orchestration – *Does the architecture support orchestration of multi-step processes involving AI components?*

Criterion & Description	Yes/No/Partial	Comments
Cross-Component Coordination – Is there support for coordinating sequences of tasks between the AI system, other systems, and human inputs? *For complex workflows (e.g., an AI model triggers an action in another system, then a human review, etc.), the architecture might use an orchestration engine or well-defined process to manage state and sequence.*		
Use of Orchestration Tools – Does the design leverage workflow engines or orchestration platforms (e.g., Airflow, Argo, etc.) where appropriate? *Explicit orchestration logic (rather than ad-hoc in code) can improve maintainability. Check if such tools are used to manage multi-step AI pipelines or if the approach is hard-coded, which can be brittle.*		
State Management – For multi-step or stateful interactions (like conversational AI sessions or long-running processes), how does the architecture handle state? *Ensure there are provisions (databases, state stores, context passing mechanisms) to maintain context across interactions or workflow steps.*		
Orchestration Performance – Does the orchestration approach support scalability and performance (parallel task execution, caching intermediate results, efficient resource use)? *Well-architected workflows should allow concurrent operations when possible and reuse results to avoid redundancy. Evaluate if the system can scale orchestrations without bottlenecks.*		

Criterion & Description	Yes/No/Partial	Comments
Advanced Workflow Support – Can the architecture integrate with Business Process Automation tools or manage multiple AI agents in a workflow ("AI teaming")? *This checks for forward-looking flexibility – e.g., the ability to orchestrate complex processes beyond the AI system itself, coordinating with external automation tools or multiple AI components working together.*		

Responsible AI Architecture

*This dimension examines whether **ethical, trustworthy, and secure AI practices** are embedded into the architecture by design. It aligns with the "Responsibility by Design" principle and looks for architectural support of AI risk management frameworks (e.g., NIST AI RMF, ISO/IEC AI standards) to ensure the system operates securely, transparently, and fairly.*

Security & Resilience – *How does the architecture protect against threats and ensure reliable operation under stress?*

Criterion & Description	Yes/No/Partial	Comments
Protection of AI Assets – Are there controls to protect AI models and data from unauthorized access or tampering? *This includes securing model files, data at rest (encryption), data in transit, and using techniques like confidential computing to protect data in use.*		
Secure Development Practices – Does the architecture and development process integrate security best practices (secure SDLC, regular vulnerability scanning, penetration testing, adversarial testing of AI models)? *Assess if security is considered throughout development – e.g., code reviews focusing on security, testing AI behavior against adversarial inputs, and quick patching of discovered vulnerabilities.*		

Criterion & Description	Yes/No/Partial	Comments
Robustness – Are architectural patterns in place to make the AI system robust to malformed or unexpected inputs and shifting data distributions? *For instance, input validation, anomaly detection on model inputs/outputs, and fallback behaviors contribute to robustness. The system should detect when inputs are out-of-scope and handle them safely.*		
Resilience – Can the system **withstand and recover** from failures or attacks gracefully? *Evaluate if the architecture provides fail-safes or graceful degradation modes (the system doesn't catastrophically fail). Also consider defenses against AI-specific threats like **prompt injection** in LLMs, data poisoning of training data, or model evasion attacks.*		

Privacy – *How does the architecture uphold data privacy and compliance?*

Criterion & Description	Yes/No/Partial	Comments
Privacy-Enhancing Technologies (PETs) – Does the architecture support techniques to preserve privacy (e.g., differential privacy, federated learning, homomorphic encryption, secure multi-party computation)? *These **PETs** (see Definitions) can allow the system to use or share data insights without exposing sensitive information.*		

Criterion & Description	Yes/No/Partial	Comments
Privacy by Design & Compliance – Is compliance with privacy regulations (GDPR, CCPA, etc.) facilitated by the architecture? *Check for features like data minimization, segregation of personally identifiable information (PII), and whether privacy considerations are built into each component. Evidence of **Privacy by Design** might include consent management, anonymization, or easy data deletion to meet regulatory requirements.*		
User Data Rights – Does the system design support fulfilling **Data Subject Access Requests (DSARs)** or similar user rights requests efficiently? *For example, can the organization easily retrieve, export, or delete a user's data across the system if requested? Efficient mechanisms for DSARs indicate good governance of user data.*		

Fairness & Bias Mitigation – *How does the architecture address potential bias and fairness issues in AI outputs?*

Criterion & Description	Yes/No/Partial	Comments
Bias Monitoring – Are there components or logging in place to monitor model inputs and outputs for potential bias against protected groups? *The system should have hooks to measure and detect bias (e.g., bias metrics, disparate impact analysis) possibly by integrating external fairness assessment tools or dashboards.*		

Criterion & Description	Yes/No/Partial	Comments
Intervention Mechanisms – Does the architecture support interventions (retraining models, recalibrating outputs) if bias is detected? *This could include design features like modular model pipelines where a biased model can be retrained or replaced, or configuration to adjust decision thresholds, etc., as part of maintenance when fairness issues are identified.*		
Diverse Data & Testing – Does the architecture facilitate using diverse, representative datasets to mitigate bias, and are there processes to test for fairness? *For example, support for feeding in additional data from under-represented groups, or simulation testing with different demographic data. A robust design acknowledges bias risks and allows improvements via data management.*		

Transparency & Explainability – *Are the AI system's workings transparent, and can it explain its decisions?*

Criterion & Description	Yes/No/Partial	Comments
Logging & Auditing – Does the architecture include comprehensive logging of data flows, model decisions, and user interactions? *Such logs are crucial for traceability and auditing of AI decisions. They should record key events (data received, model inference results, actions taken) for later review.*		
Explanation Mechanisms – Are there provisions to generate explanations for the AI's outputs or decisions when needed? *Examples include storing additional metadata to explain a recommendation, using explainable AI techniques (like SHAP or LIME) integrated into the system, or having a simpler surrogate model for explanation purposes.*		

Criterion & Description	Yes/No/Partial	Comments
System Transparency – Is information about the AI system's capabilities, limitations, and use communicated clearly (e.g., documentation or user-facing disclosures)? *A good architecture supports capturing and exposing metadata about the model (training data sources, intended usage, accuracy) and aligns with transparency standards or guidelines.*		

Accountability & Governance – *Does the architecture facilitate oversight and governance of the AI system?*

Criterion & Description	Yes/No/Partial	Comments
Human Oversight & Control – Are there mechanisms for human reviewers to oversee, intervene, or override AI decisions when necessary? *For higher-risk AI functions, the system should include checkpoints or controls (e.g., a human must approve certain outputs, or an emergency "off switch" to halt the AI) to ensure ultimate human control.*		
Roles & Responsibilities – Does the architecture define or enforce clear roles for those who develop, deploy, and maintain the AI (e.g., who can update models, who monitors outcomes)? *Clarity in responsibility (see **RBAC** in Definitions) is often implemented via access controls and process checkpoints in the system – for example, only authorized ML engineers can push model updates, and governance officers receive alerts for policy violations.*		

Alignment with Governance Frameworks – Are the system's design choices in line with organizational AI policies or external standards (e.g., NIST AI Risk Management Framework, ISO/IEC 42001)? *There should be evidence that best-practice guidelines for AI governance are considered. For instance, if the organization has an AI ethics policy, the architecture includes features to enforce it (like an ethics checklist in model deployment).*		

Safety – *How does the architecture ensure the AI system operates safely and mitigates potential harm?*

Criterion & Description	Yes/No/Partial	Comments
Fail-Safe Design – Does the system have fail-safe mechanisms to prevent harm if it malfunctions or encounters an unknown scenario? *For example, if the AI cannot handle a situation, it should default to a safe state or hand off to a human. Identify patterns like circuit breakers, fallback defaults, or conservative limitations that prevent unsafe outcomes.*		
Emergency Control Mechanisms – Can operators quickly disengage or shut down the AI system in an emergency or if it behaves unexpectedly? *This could be as simple as a manual "kill switch" or as complex as automated monitoring that triggers shutdown under certain conditions. Alignment with relevant safety standards (like IEEE 7010 for AI safety) is a good sign.*		

Compliance Readiness – *Does the architecture support compliance with regulations and standards in the AI's domain?*

Criterion & Description	Yes/No/Partial	Comments
Regulatory Adherence – Is the system designed to help meet applicable AI-related regulations or industry-specific laws (e.g., EU AI Act, healthcare AI regulations)? *Check for features that facilitate compliance – for instance, if in healthcare, does the architecture address HIPAA requirements? There should be evidence of considering legal constraints in the design.*		
Evidence Generation for Audits – Does the architecture support generating the documentation and logs needed for compliance audits or reporting? *This includes maintaining records of model versions, data sources, decisions made, and changes over time, in a way that an auditor could review. Consider if the system can easily output an audit report or if everything is manually collected.*		

Capability Enablement & Value Realization

This final dimension checks whether the architecture effectively translates technical capabilities into business or user value. It ties back to being "Capability-Centric" and practicing "Pragmatism & Domain Specificity" – meaning the design is driven by real-world goals and constraints. We assess how well the architecture meets performance needs, scales, and remains maintainable to continuously deliver value.

Goal Alignment – *Is the architecture aligned with the intended business or mission goals of the ACI system?*

Criterion & Description	Yes/No/Partial	Comments
Business Objectives Support – Does the architecture directly support the key business objectives or use-case goals (e.g., improving efficiency, decision support, customer experience)? *There should be a clear connection (a "line of sight") from architectural components to the value they provide. For example, if the goal is faster customer service, the architecture includes components (like an AI assistant integrated into a CRM) that enable that outcome.*		
Task/Function Enablement – Is the architecture well-suited to perform the specific tasks the ACI system is intended for? *Evaluate if any architectural choices hinder the AI's core functions. The design should make it easy to implement the required AI capabilities (e.g., the pipeline for data->model->result is streamlined for the tasks at hand).*		
User Needs & UX – Does the architecture consider end-user needs and experience (UI/UX)? *For instance, if end-users interact with the AI's output, does the system provide timely responses and the necessary data? A user-centric architecture might include fast response times, user feedback loops, or interfaces that make the AI's output accessible.*		

Performance & Scalability – *Can the system meet performance requirements and scale to future demands?*

Criterion & Description	Yes/No/Partial	Comments
Performance Targets – Is the architecture capable of meeting required performance metrics (throughput, latency, response time)? *Assess whether any component is likely to be a bottleneck. For example, if real-time responses are needed, is the model hosting environment low-latency? Look for use of optimized data structures, caching, or high-performance compute where appropriate.*		
Scalability Patterns – Does the design include patterns for scalability, such as microservices, load balancing, horizontal scaling, etc.? *The system should be able to handle increased load by scaling out or up. Check for cloud scalability (auto-scaling groups, container orchestration) or modular design that allows distribution of workload. Efficient database design and stateless service design are good signs.*		
Resource Optimization – Is the architecture optimized for efficient resource use (CPU, GPU, memory, network) to manage cost and performance? *For example, does it use batch processing where real-time isn't needed, or schedule heavy jobs during off-peak times? Efficient use of hardware (like using GPUs for ML inference only when necessary) indicates the system can deliver value cost-effectively.*		

Reliability & Maintainability – *Does the architecture support reliable operations and ease of maintenance over time?*

Criterion & Description	Yes/No/Partial	Comments
Fault Tolerance – Are there features that ensure the system can tolerate failures (e.g., redundant components, graceful degradation, health checks)? *Check if critical services are replicated, if failover mechanisms exist (for example, a standby model server if the primary fails), and if the system can continue partial functionality when parts fail.*		
Monitoring & Observability – Does the architecture have monitoring and logging across all components to detect issues proactively? *There should be telemetry (metrics, logs, alerts) for infrastructure, data pipeline health, model performance, etc. Observability is key for reliability – e.g., integration with monitoring tools (Prometheus, ELK stack) to quickly identify anomalies.*		
MLOps & Deployability – For ML-centric systems, does the architecture support **MLOps** practices (automated model testing, CI/CD for model updates, model versioning, rollbacks)? *Frequent model updates require an infrastructure that can deploy new models reliably without breaking the system. Look for CI/CD pipelines, containerization of models, or use of model registries. Strong MLOps capability enables rapid iteration while maintaining stability.*		

Criterion & Description	Yes/No/Partial	Comments
Long-term Maintainability – Is the system built with maintainability in mind (modular codebase, well-defined interfaces, good documentation, etc.)? *Consider how easy it is to modify or extend the system. Are components loosely coupled and clearly interfaced (so changes in one don't cascade failures)? High-quality documentation and coding standards also contribute to maintainability.*		

Reporting Phase

In the Reporting phase, compile the insights from the evaluation into a clear summary and actionable recommendations. This section serves to document the **findings**, highlight strengths and weaknesses, and propose **improvements** to enhance the ACI architecture's alignment with best practices.

Findings Summary

(Summarize the key findings of the evaluation. Outline the architecture's strengths, any weaknesses or gaps identified, and note areas of non-compliance with the evaluation criteria or ACI principles.)

- *Strengths:* e.g., well-implemented data governance, strong integration with enterprise systems, etc.
- *Weaknesses/Gaps:* e.g., missing bias monitoring, performance bottleneck in data pipeline, etc.
- *Risks:* e.g., potential security vulnerabilities, reliability issues under peak load, etc.

Use the space above to detail the outcome of the evaluation. For instance, list specific strengths (architectural choices that are exemplary or exceed requirements) and specific weaknesses or gaps. Include any risks these weaknesses pose (e.g., "Lack of X could lead to Y risk").

Recommendations

(List actionable recommendations to address the identified issues or to further improve the architecture. Prioritize these based on impact and urgency.)

- *Recommendation 1:* … (e.g., implement an API gateway to improve security and monitoring of integrations)
- *Recommendation 2:* … (e.g., incorporate bias detection tools into the model pipeline and retrain model on more diverse data)
- *Recommendation 3:* … (e.g., improve documentation and logging for better transparency and auditability)

Each recommendation should be clear and achievable – include what should be done and why. Also consider the priority (e.g., High, Medium, Low) based on how critical the issue is to the system's goals and risks.

Review Phase

The Review phase involves final validation, approval of the evaluation results, and plans for continuous improvement. Ensure that the evaluation report is reviewed by the appropriate authority and that a plan is in place to iterate on this process.

Review Checklist ***(Post-evaluation wrap-up)***	**Yes/No**	**Comments**
Formal Approval Completed – Has the evaluation report been reviewed and approved by the designated approver or governance body? *Confirm that the responsible authority (e.g., an AI governance board or project sponsor) has formally signed off on the findings and recommendations.*		

Note*:* It's recommended to **capture lessons learned** from this evaluation and update the evaluation criteria or process for next time. ACI architecture evaluation should be an ongoing process – consider scheduling a follow-up review or integrating these checks into the project lifecycle (e.g., at each major release). Continuous improvement ensures the architecture remains aligned with evolving best practices and requirements over time.

Definitions

Key terms and abbreviations used in this template:

Artificial Capable Intelligence (ACI)

- AI systems designed to deliver practical, goal-oriented capabilities **effectively, reliably, and responsibly within a specific domain**. ACI is more advanced than narrow AI (single-task systems) but is not as broad as a hypothetical general AI; it focuses on tangible, domain-specific intelligence and value.

Capability-Centric

- An ACI architecture principle stating that the design's primary purpose is to **enable the specific tasks and goals** the AI system is meant to achieve. Every component should demonstrably contribute to the system's intended capabilities.

Pragmatism & Domain Specificity

- An ACI principle that the architecture be **grounded in practical application** for its defined context, favoring effective real-world solutions over theoretical perfection. It should avoid unnecessary generality outside its domain scope.

Data as Fuel

- An ACI principle recognizing that **high-quality, well-governed data** is fundamental to AI performance. The architecture must efficiently ingest, manage, and utilize data, ensuring integrity and accessibility of data throughout its lifecycle.

Integration First

- An ACI principle that the system **must integrate seamlessly** with existing enterprise systems and workflows. Rather than functioning in isolation, the AI should connect with other components and data sources to deliver end-to-end value.

Responsibility by Design

> An ACI principle that ethical and responsible AI practices (security, privacy, fairness, transparency, compliance) **must be embedded into the architecture from the outset**, not added on later. This ensures the system inherently supports trustworthy AI operation.

Privacy-Enhancing Technologies (PETs)

Tools and techniques that enable useful data analysis or AI modeling **without compromising privacy**. Examples include differential privacy, federated learning, homomorphic encryption, and secure multi-party computation. These help protect sensitive data in AI systems.

Data Subject Access Request (DSAR)

A request from an individual to an organization asking for access to the personal data that the organization has collected about them (What is a DSAR (Data Subject Access Request)?). Regulations like GDPR grant people the right to such requests, and AI architectures should be able to accommodate retrieving or deleting an individual's data on demand.

Role-Based Access Control (RBAC)

A security model where access permissions are **assigned to roles** (e.g., user, admin, reviewer) rather than to individuals directly. Users are granted roles, and through those roles they acquire permissions to systems and data (What Is Role-Based Access Control (RBAC)? | IBM). In an ACI context, RBAC ensures that only authorized roles (such as an Approver) can perform certain actions like approving an evaluation or accessing sensitive data.

MLOps (Machine Learning Operations)

A set of practices that apply DevOps-like principles to the ML lifecycle, aiming to **streamline deploying and maintaining machine learning models** in production (What is MLOps?). MLOps encompasses continuous integration and deployment of models, automated testing, monitoring of model performance, and data/version management to keep the AI system reliable and up-to-date.

Cited References

1. **F. Nolasco**, *"A Methodology for Evaluating Artificial Capable Intelligence (ACI) Architectures (Platform-Agnostic),"* May 2025. (Content used for evaluation dimensions and criteria)

2. **Usercentrics**, *"What is a data subject access request (DSAR)? How-to guide,"* accessed 2025 (What is a DSAR (Data Subject Access Request)?). (Definition of DSAR)

3. **IBM**, *"What is role-based access control (RBAC)?",* IBM Think Blog, Aug. 2024 (What Is Role-Based Access Control (RBAC)? | IBM). (Definition of RBAC)

4. **Red Hat**, *"What is MLOps?",* Sep. 26, 2023 (What is MLOps?). (Definition of MLOps and its practices)