ARCO Technical Overview

Deterministic Regulatory Classification for AI Systems

Assurance & Regulatory Classification Ontology

1. The Problem ARCO Addresses

Modern AI regulation evaluates systems based on **capability**, not configuration. Under frameworks like the EU AI Act, a system may trigger regulatory classification because of what it is structurally capable of doing—regardless of whether that capability is currently enabled, deployed, or intended for use.

This creates a specific problem: traditional compliance approaches operate downstream. Teams build systems, train models, and deploy pipelines—then ask whether what they have built is acceptable under regulation. At that point, compliance becomes reactive. Risk is explained after the fact. Documentation is produced to justify decisions already made.

The cost of this ordering is substantial. A misclassification discovered post-deployment can escalate from a five-figure compliance concern to a seven-figure remediation problem. Retraining, redesign, delayed launches, regulatory fines, and forced suspension compound rapidly once a system is in production.

Existing tools do not solve this problem adequately:

- **Static documentation** (PDFs, spreadsheets, model cards) cannot capture inherited capabilities or conditional system relationships. Text-based reviews cannot reliably map system components to regulatory definitions.
- **Self-attestation checklists** depend on manual interpretation without systematic verification. They describe what is configured, not what is structurally possible.
- **Probabilistic AI tools** (RAG systems, embedding-based classifiers) optimize for relevance, not admissibility. They produce confidence scores, not logical entailment. In regulatory contexts, a near-match is not 'almost correct'—it is categorically wrong.

The fundamental gap is between *interpretation* and *determination*. Interpretation asks what a system might resemble. Determination asks what a system *is*, under a formally grounded regulatory ontology, and whether that triggers specific classification criteria.

ARCO exists to close that gap: to move regulatory risk decisions upstream, before investment is sunk, and to produce those decisions through formal reasoning rather than probabilistic approximation.

2. ARCO's Core Claim

ARCO provides **deterministic regulatory classification** for AI systems.

When ARCO determines that a system is High-Risk under Article 6 and Annex III of the EU AI Act, that determination is not a prediction, a confidence score, or an advisory opinion. It is a **logical conclusion** derived from explicitly encoded facts about the system, explicitly represented regulatory criteria, and formal inference over both.

This has three immediate consequences:

- **Reproducibility.** Any party with access to the same ontology, instance data, and reasoning artifacts can independently verify the determination. The conclusion does not depend on opaque model weights or undocumented heuristics.
- **Auditability.** Every step in the reasoning chain is inspectable. If a determination is disputed, the disagreement can be traced to specific premises—either the system was encoded incorrectly, the

regulatory criteria were misrepresented, or the logic itself is contested. There is no hidden layer.

- **Defensibility.** The determination can be presented to regulators, auditors, or procurement reviewers as a structured justification, not a black-box output. The reasoning is legible to humans and machine-checkable.

ARCO operates at the design stage, before deployment decisions are finalized. It answers a specific question: *Given the system's structural capabilities, does it satisfy the necessary and sufficient conditions for a given regulatory classification?*

This is not a claim to eliminate regulatory risk or to substitute for legal judgment. It is a claim to make the basis for classification explicit, structured, and formally justified—so that humans can make informed decisions with clear visibility into the reasoning.

3. Architectural Overview

ARCO's architecture reflects a deliberate separation of concerns. Information extraction, ontological representation, and logical reasoning are handled by distinct components with different epistemic properties. This separation is not incidental—it is the mechanism by which ARCO maintains the boundary between probabilistic interpretation and deterministic judgment.

3.1 Neuro-Symbolic Separation

The first architectural principle is that language models are used only at the boundary between unstructured text and structured representation.

In practice, this means:

- LLMs may be used to extract candidate capabilities from system documentation, technical specifications, policy text, or design artifacts
- LLM outputs are treated as *proposals*, not conclusions
- Whether a proposed capability is accepted and asserted in the ontology remains a human decision

LLMs never touch compliance conclusions directly. They do not classify systems. They do not evaluate regulatory criteria. They do not produce determinations.

This design reflects a clear epistemic boundary. Language models are useful for interpretation—identifying relevant passages, surfacing potential capabilities, organizing unstructured information. But interpretation is not judgment. The compliance determination must be grounded in explicit facts and formal inference, not statistical pattern matching.

3.2 Ontological Grounding

The second architectural principle is that all compliance-relevant entities are represented in a formally grounded ontology aligned with Basic Formal Ontology (BFO) and extended with Common Core Ontologies (CCO) and Information Artifact Ontology (IAO).

Reality-side entities (systems, capabilities, processes) are modeled using BFO categories:

- **Systems** are modeled as BFO Object Aggregates—material entities composed of parts

- **Capabilities** are modeled as BFO Dispositions—realizable entities that inhere in their bearers independent of whether they are currently realized
- **Operational processes** are modeled as BFO Processes that realize dispositions and have participants

Representation-side entities (regulatory content, documentation, determinations) are modeled using IAO and CCO:

- **Regulatory content** (provisions, conditions, criteria) are Information Content Entities that are *about* reality-side universals
- **Assessment documentation** is an Information Output produced by documentation processes
- **Compliance determinations** are structured results that are *about* both systems and regulatory content

This distinction matters. The ontology does not collapse legal text into executable rules. It represents what regulatory provisions *say* (as information content) and what they are *about* (capability classes). The link between a system and a regulatory classification is established through formal relations, not string matching or keyword extraction.

3.3 Validation and Reasoning Pipeline

The third architectural principle is that validation and reasoning are performed by deterministic, inspectable mechanisms—not by learned models or opaque inference.

SHACL (Shapes Constraint Language) enforces structural completeness before reasoning begins. SHACL operates under closed-world assumptions, meaning it can detect *missing* information rather than inferring it. If a system representation lacks required components—if documentation is not linked to a system, if a provider role is not linked to an organization—SHACL validation fails explicitly.

This prevents incomplete or malformed data from producing spurious classifications. The closed-world validation layer ensures that every determination rests on a structurally complete foundation.

OWL (Web Ontology Language) defines class restrictions and equivalence axioms that enable logical inference. When a system bears a disposition that instantiates a regulated capability class, and regulatory content is defined as being *about* that capability class, the classification follows by OWL entailment.

The reasoner does not guess. It computes logical consequences from the asserted facts and the ontology's axioms. If the premises are satisfied, the conclusion holds necessarily.

SPARQL ASK queries provide deterministic criteria testing and audit traceability. ASK queries return boolean values—TRUE or FALSE—with no ambiguity. They test whether specific patterns exist in the knowledge graph, producing machine-readable results suitable for compliance logs and regulatory submission.

Together, these three layers form a pipeline:

1. **Extraction** (neuro): LLMs surface candidates from unstructured sources
2. **Representation** (symbolic): Human-validated candidates are encoded in the BFO-aligned ontology
3. **Validation** (SHACL): Structural completeness is enforced
4. **Reasoning** (OWL): Logical consequences are computed
5. **Audit** (SPARQL): Criteria satisfaction is tested and logged

Each layer has a distinct role and a distinct epistemic status. Probabilistic tools assist with recall; the ontology provides precision; formal reasoning provides proof.

The output of this pipeline is not a prediction. It is a justified conclusion, traceable from regulatory criteria through system representation to final determination.

4. Worked Example: Sentinel-ID Regulatory Determination

To demonstrate how ARCO operates in practice, this section walks through a complete regulatory classification using Sentinel-ID, a synthetic reference system designed to validate the reasoning pipeline.

4.1 Why a Synthetic Reference System

Real AI systems present significant obstacles to transparent compliance analysis. Hardware specifications are often proprietary. System documentation is incomplete or inconsistent. Compliance-relevant data is not publicly available. These constraints make it impractical to demonstrate reasoning correctness using production systems in an open or academic setting.

Sentinel-ID exists to solve this problem. It is not a guess about a real system or a simplified toy example. It is a controlled reference system with fully observable components, explicitly modeled capabilities, and complete documentation. Every element of Sentinel-ID is designed to be inspectable, allowing the compliance reasoning pipeline to be validated with complete transparency.

The focus is on reasoning correctness, not vendor claims. If the logic works on Sentinel-ID, the same logic applies to any system that can be encoded in the same ontological structure.

4.2 System Description

Sentinel-ID is modeled as an AI-enabled system with material components that bear specific capabilities. The system includes hardware and software elements that enable:

- Capture of human facial features
- Extraction of biometric templates
- Comparison against stored identity representations

These capabilities exist at the structural level. They are properties of the system by virtue of its components, independent of whether any particular capability is currently enabled in software configuration.

In ontological terms, Sentinel-ID is represented as an instance of :System, which is a subclass of BFO Object Aggregate. The system bears a disposition—:Sentinel_FaceID_Disposition—which is an instance of :BiometricIdentificationCapability. This disposition inheres in the system regardless of operational state.

```
:Sentinel_ID_System rdf:type :System ;
  rdfs:label "Sentinel-ID System" ;
  ro:0000053 :Sentinel_FaceID_Disposition .  # bearer of

:Sentinel_FaceID_Disposition rdf:type :BiometricIdentificationCapability ;
  rdfs:label "Sentinel Facial Recognition Disposition" .
```

The key modeling decision is that capability is represented as disposition, not behavior. The system does not need to be actively performing biometric identification to bear the capability. This reflects how regulatory frameworks actually evaluate risk: based on what a system *can* do, not merely what it *is* doing.

4.3 Regulatory Mapping

The EU AI Act, specifically Article 6 and Annex III, classifies AI systems as High-Risk based on their capabilities and deployment contexts. Annex III enumerates specific categories, including systems intended for biometric identification of natural persons.

In ARCO, regulatory provisions are not encoded as executable rules or procedural logic. They are represented as Information Content Entities—structured representations of what the law *says*, not automated implementations of what the law *does*. This preserves the distinction between legal text (which requires interpretation) and compliance reasoning (which operates over explicit criteria).

```
:AnnexIII_Condition_Q1 rdf:type :RegulatoryContent ;
  rdfs:label "Annex III Q1 (Biometric Rule)" ;
  iao:0000136 :BiometricIdentificationCapability .  # is about
```

The critical relation here is iao:0000136 (is about). The regulatory content :AnnexIII_Condition_Q1 is *about* the universal :BiometricIdentificationCapability. This creates a formal link between the regulatory provision and the capability class.

When an individual system bears a disposition that instantiates this capability class, the regulatory provision applies to that system—not by assertion, but by logical structure.

4.4 Reasoning Trace

The determination proceeds through three stages: structural validation, criteria testing, and inference.

Stage 1: Structural Validation (SHACL)

Before any regulatory reasoning occurs, SHACL constraints verify that the system representation is structurally complete. The shapes enforce requirements such as:

- Assessment documentation must be linked to both a System and a RegulatoryContent entity
- The assessment documentation process must have participants (System, ProviderRole) and outputs (AssessmentDocumentation)
- ProviderRole must inhere in a ProviderOrganization

If any required structure is missing, validation fails and the system cannot proceed to classification. This prevents 'garbage in, true out' scenarios where incomplete data silently passes through reasoning.

Stage 2: Criteria Testing (SPARQL ASK)

With structural validity confirmed, SPARQL ASK queries test whether the system satisfies the specific conditions defined for regulatory classification.

```
PREFIX : <https://arco.ai/ontology/core#>
PREFIX iao: <http://purl.obolibrary.org/obo/IAO_>

ASK WHERE {
  :AssessmentDoc_001 iao:0000136 :Sentinel_ID_System .
  :AssessmentDoc_001 iao:0000136 :AnnexIII_Condition_Q1 .
}
```

This query asks: Does the assessment documentation link the system to the relevant Annex III condition?

Result: **TRUE**

Stage 3: Logical Inference (OWL)

The SPARQL result confirms that the documentation structure links Sentinel-ID to the biometric identification condition. But the *classification* itself—the determination that Sentinel-ID is a High-Risk system—follows from OWL reasoning over the ontology.

The reasoning chain is:

1. Sentinel-ID is a :System
2. Sentinel-ID bears (ro:0000053) a disposition that is an instance of :BiometricIdentificationCapability
3. :AnnexIII_Condition_Q1 is about (iao:0000136) the class :BiometricIdentificationCapability
4. Therefore, the system instantiates a capability that falls under Annex III criteria
5. Under Article 6, systems falling under Annex III criteria are classified as High-Risk

The conclusion is not asserted in the instance data. It is *entailed* by the ontological structure. This is the core distinction: the ontology logically forces the determination rather than the author asserting it.

Final Determination: HIGH-RISK AI SYSTEM

This determination is:

• Deterministic (follows from premises by logical necessity)

• Reproducible (any reasoner with the same inputs produces the same output)

• Auditable (every step is inspectable)

• Independent of discretionary judgment after encoding

4.5 Governance Traceability

The determination does not exist in isolation. ARCO's governance extension ensures that every classification is traceable to a responsible organizational entity.

The traceability chain for Sentinel-ID:

```
ProviderOrg_001 (ProviderOrganization)
    has role (ro:0000087) --> ProviderRole_001 (ProviderRole)
                                  inheres in (ro:0000052) --> ProviderOrg_001

AssessmentDocProcess_001 (AssessmentDocumentationProcess)
    has participant (ro:0000057) --> Sentinel_ID_System
    has participant (ro:0000057) --> ProviderRole_001
    has output (cco:has_output) --> AssessmentDoc_001

AssessmentDoc_001 (AssessmentDocumentation)
    is about (iao:0000136) --> Sentinel_ID_System
    is about (iao:0000136) --> AnnexIII_Condition_Q1
```

This structure answers the question: Who is responsible for this system, and how was the compliance determination produced?

The Provider Organization bears the Provider Role. The Provider Role participates in the Assessment Documentation Process. That process produces Assessment Documentation that is formally linked to both the system and the regulatory content. Every link is explicit, typed, and queryable.

This is not metadata. It is part of the formal compliance record, suitable for audit, regulatory submission, and liability analysis.

5. What ARCO Produces

An ARCO engagement produces three primary artifacts.

Regulatory Determination Certificate

A formal statement of regulatory classification derived from the system's structurally encoded capabilities and the applicable regulatory criteria. The certificate identifies:

- The final classification (e.g., High-Risk, Limited-Risk, Out of Scope)
- The specific capabilities that triggered the classification
- The regulatory provisions that applied

This is not an advisory opinion. It is a documented conclusion supported by explicit reasoning.

Traceability Log

A machine-readable audit artifact generated directly from the validation and reasoning pipeline. The log includes:

- SHACL validation results confirming structural admissibility
- SPARQL ASK query outputs demonstrating criteria satisfaction
- The inference chain linking system properties to regulatory conclusions

This artifact enables independent reproduction. Any party with access to the ontology, instance data, and query artifacts can verify the determination without relying on discretionary judgment.

Gap Analysis Report

A precise enumeration of the system components, encoded capabilities, and structural features that instantiated the regulatory classification conditions. This report is expressed in ontological terms, mapping directly to the formal representation used in reasoning.

The gap analysis is not a remediation plan. It identifies *what* triggered the classification, not *how* to change it. Remediation decisions remain with the system owner and their legal counsel.

6. Scope and Limitations

ARCO is designed to do one thing well: produce defensible regulatory classifications based on explicit system structure and formal reasoning. It is not designed to do everything.

What ARCO does:

- Establishes regulatory classification through deterministic inference
- Makes the structural basis for classification explicit and inspectable
- Produces audit-ready artifacts suitable for regulatory submission

- Enables pre-deployment classification decisions

What ARCO does not do:

Recommend mitigation. ARCO identifies classification; it does not prescribe remediation. What to do about a High-Risk classification is a business and legal decision, not an ontological one.

Assess proportionality or exemptions. Regulatory frameworks often include provisions for exemptions, proportionality considerations, or context-dependent adjustments. These require legal interpretation beyond formal classification.

Substitute for legal counsel. ARCO produces technical determinations. Legal advice requires licensed professionals who can account for jurisdiction, precedent, and strategic considerations that formal reasoning cannot capture.

Automate legal judgment. The determination follows from explicitly encoded facts and criteria. The encoding itself—deciding what facts to assert, how to represent system capabilities, which regulatory provisions apply—requires human judgment.

Eliminate the need for human oversight. ARCO makes assumptions visible and reasoning explicit. It does not remove humans from the loop; it gives them a clearer basis for decision-making.

These boundaries are features, not limitations. A system that claimed to automate legal judgment or eliminate human oversight would be less credible, not more. ARCO's value lies precisely in making the reasoning transparent enough that humans can exercise informed judgment over it.

7. Engagement Model

ARCO is delivered as a structured assessment, not an open-ended consulting engagement or a software license.

Scope

A standard engagement covers one AI-enabled system. The system boundary is fixed at the outset: what components are included, what documentation is available, what deployment context applies. No implicit assumptions are introduced during the assessment.

Execution Phases

The engagement proceeds through defined phases:

1. **Ingestion and Boundary Definition** — System documentation is mapped to explicit ontological instances. The system boundary is established.
2. **Structural Validation** — SHACL constraints enforce documentation completeness. Missing or underspecified elements are surfaced explicitly.
3. **Deterministic Classification** — OWL reasoning and SPARQL queries evaluate the system against regulatory criteria. The classification is derived, not asserted.
4. **Determination and Handover** — Final artifacts are produced and delivered. The determination certificate, traceability log, and gap analysis constitute the formal output.

Exclusions

An ARCO engagement does not include:

- Remediation or system redesign
- Legal interpretation or advice
- Access to a hosted software platform
- Ongoing monitoring or continuous assessment

The engagement produces a point-in-time determination based on the system as documented. Changes to the system after assessment would require re-evaluation.

Applicability

This engagement model is designed for organizations that need a clear, defensible answer to a specific regulatory classification question before committing to deployment. It is particularly relevant for:

- Systems with potential High-Risk classification under EU AI Act Annex III
- Pre-deployment regulatory readiness assessments
- Internal governance decisions requiring formal justification
- Procurement or audit contexts requiring reproducible compliance evidence

ARCO does not require access to model weights, training data, or runtime behavior. It operates on system documentation, architecture specifications, and capability declarations. This makes it applicable to systems at the design stage, before operational data exists.

Appendix: Technical Reference

This appendix provides reference information for readers who wish to inspect the underlying artifacts.

Namespace Prefixes

Prefix	URI
:	https://arco.ai/ontology/core#
bfo:	http://purl.obolibrary.org/obo/BFO_
ro:	http://purl.obolibrary.org/obo/RO_
iao:	http://purl.obolibrary.org/obo/IAO_
cco:	http://www.ontologyrepository.com/CommonCoreOntologies/

Key BFO/RO/IAO Relations

Relation	URI	Usage
bearer of	ro:0000053	System bears Disposition
inheres in	ro:0000052	Role inheres in Organization
has participant	ro:0000057	Process has participant

has role	ro:0000087	Organization has Role
realizes	bfo:0000055	Process realizes Disposition
has part	bfo:0000051	Composite has part
is about	iao:0000136	Information Content is about entity
has output	cco:has_output	Process has output

Core ARCO Classes

Class	Parent	Description
:System	bfo:0000027	AI system under evaluation
:CapabilityDisposition	bfo:0000016	System capability
:BiometricIdentificationCapability	:CapabilityDisposition	Regulated capability type
:RegulatoryContent	cco:InformationContentEntity	Regulatory provision
:ComplianceDetermination	cco:InformationContentEntity	Classification result
:AssessmentDocumentation	:InformationOutput	Documentation artifact

Artifact Locations

For technical review, the following artifacts constitute the ARCO reference implementation:

- **ARCO_core.ttl** — Core ontology definitions
- **ARCO_governance_extension.ttl** — Governance and documentation classes
- **ARCO_instances_sentinel.ttl** — Sentinel-ID instance data
- **assessment_documentation_shape.ttl** — SHACL validation shapes
- **check_assessment_traceability.sparql** — Criteria testing query
- **run_pipeline.py** — Execution pipeline script