A Whitepaper on Structured Ethical Facts, Modular Governance, and MCP Integration for ErisML

Status: Draft
Version: 0.3 (DEME Profiles & MCP Update)
Date: 2025-12-12

Author: Andrew Bond
andrew.bond@sjsu.edu

Abstract

The ErisML Vision Paper proposes democratically governed Ethical Modules (EMs) for autonomous agents: multiple stakeholders encode their values into distinct modules whose "votes" are aggregated by a governance layer. DEME (Democratically Governed Ethics Modules) is the concrete realization of that idea.

This whitepaper refines and extends the vision in one specific direction:

Ethics modules should do ethics only. They should reason purely over abstract ethical facts rather than raw domain data (for example, ICD codes, sensor traces, or low-level control signals).

We propose:

- A structured **EthicalFacts** abstraction: a domain-agnostic, extensible schema that captures ethically relevant dimensions (consequences; rights and duties; justice and fairness; autonomy and agency; privacy and data governance; societal and environmental impact; virtue and care; procedural legitimacy; epistemic status).
- A clean **EthicsModule** interface that implements purely normative reasoning over EthicalFacts and returns an **EthicalJudgement**.
- A separation of responsibilities between **domain and assessment layers** (which interpret raw data, compute risk/benefit, detect rights violations, and so on) and **ethics modules** (which weigh those facts according to a given value system).
- Integration with a **democratic governance layer** via **DEME profiles**—versioned governance configurations that specify stakeholder weights, veto rules, lexical priority layers, and thresholds.

- A mapping from DEME's abstractions to **MCP tools** (deme.list_profiles, deme.evaluate_options, deme.govern_decision) so that any MCP-compatible agent can obtain structured ethical oversight.

We illustrate the approach using the same style of use cases as the Vision Paper, focusing particularly on:

- **Case Study 1:** Clinical triage under resource scarcity,

and sketching how the same pattern extends to:

- **Case Study 2:** Autonomous vessel navigation in shared waters, and
- **Case Study 3:** Multi-agent urban logistics.

The goal is not to present a complete ethical theory, but a framework that is explicit about what ethical dimensions it currently models and designed to grow under democratic governance, while aligning with emerging standards such as the NIST AI Risk Management Framework.

---

1. Introduction

1.1 Motivation

As autonomous agents gain the ability to make high-impact decisions—allocating medical resources, routing autonomous vessels, coordinating fleets of robots—we need machinery that:

- Encodes ethical constraints and values in a way that is transparent, auditable, and amendable through governance processes.
- Remains modular and composable so that different stakeholders can contribute their own ethical perspectives, and no single group hard-codes all values into monolithic code or models.

The ErisML Vision Paper addresses these needs through **democratically governed Ethics Modules (EMs)**—pluggable decision modules whose outputs are aggregated into system decisions. DEME gives this architecture a concrete form, with a clear API boundary around *ethics-only* reasoning.

However, a key design risk is overloading EMs with responsibilities that belong elsewhere, including:

- Parsing domain data (for example, ICD codes, AIS tracks).

- Computing prognosis, risk, or performance.
- Predicting outcomes of actions.

This violates the Single Responsibility Principle and makes EMs harder to verify and test, less interpretable, and tightly coupled to domain models that will evolve independently.

1.2 Ethics-Only Modules

To address this, DEME enforces a strict separation: **domain intelligence lives outside EMs**. Ethics modules see only EthicalFacts—a structured bundle of ethically relevant summaries.

Concrete examples:

- Instead of seeing "ICD-10 code J18.9", EMs see values such as expected_benefit = 0.8, expected_harm = 0.2, urgency = 0.9, violates_rights = False.
- Instead of seeing raw radar tracks and COLREG details for a vessel, EMs see **probability of collision**, **distribution of risk across people and property**, and **flags for legal constraint violations** and **environmental sensitivity**.

Ethics modules are then **purely normative**: they answer *"What should we do, given these ethical facts?"*—not *"What is likely to happen?"* or *"What does this ICD code mean?"*

1.3 Contributions

This whitepaper contributes:

- A data model for **EthicalFacts** and **EthicalJudgement**, including explicit treatment of autonomy and agency, privacy and data governance, and societal and environmental impact.
- An evolvable architecture for integrating ethics-only modules into ErisML and exposing them as an **MCP server** (erisml.ethics.interop.mcp_deme_server).
- The notion of **DEME profiles** as versioned governance configurations specifying stakeholder weights, veto rules, and lexical priorities.
- A worked-out example for Case Study 1 (clinical triage) and sketches for related use cases (autonomous vessel navigation and multi-agent urban logistics).

The goal is not to present a complete ethical theory, but a practical framework that:

- Is explicit about which ethical dimensions are represented at any given time,

- Can be extended under democratic governance, and
- Supports audit and regulatory alignment (for example, NIST AI RMF govern–map–measure–manage functions).

---

2. Architectural Overview

2.1 Layered Responsibility

DEME sits in a four-layer stack that cleanly separates domain intelligence from ethical reasoning and governance.

2.1.1 Raw Domain Layer

Inputs

Real-world environment and context:

- Clinical: EHR data, ICD codes, vitals, labs, clinician notes.
- Maritime: AIS tracks, radar, sonar, charts, weather feeds.
- Urban logistics: maps, traffic feeds, sensors, orders, schedules.
- External systems and databases (hospital systems, VTS, city APIs, etc.).

Responsibilities

Interface to the real world:

- Ingest sensor streams, logs, and external data sources.
- Handle connectivity, protocols, authentication, and low-level data formats.
- Perform minimal cleaning / normalization needed for downstream use:
    - Time-sync, basic validation, normalization of units and formats.

Outputs

DomainRawState:

- Time-aligned, cleaned representations of: patients, vessels, vehicles, infrastructure, environment, etc.
- Still “domain-shaped” (ICD codes, AIS messages, traffic events), but normalized and queryable.

Optionally: CandidateOptions (raw) for higher layers—for example, feasible treatment, route, or scheduling options proposed by planners/controllers.

2.1.2 Domain & Assessment Layer

Inputs

- DomainRawState from the Raw Domain Layer.
- Any CandidateOptions (raw) proposed by planners/controllers.
- Domain-specific models and rules: prognostic models, risk models, legal/regulatory rule sets, privacy policies, environmental models, etc.

Responsibilities

Interpret raw domain data into domain-level quantities, such as:

- *Clinical:* severity, prognosis with/without treatment, likelihood of benefit, likelihood of harm, urgency.
- *Maritime:* collision probability, closest point of approach, fuel usage, environmental impact.
- *Urban logistics:* travel time, congestion impact, risk to pedestrians, service-level impact.

Detect legally, technically, or policy-relevant conditions:

- Presence or absence of valid consent.
- COLREG violations, traffic rule violations.
- Licensing/credential issues, hospital or port policies.

Compute ethically relevant indicators for:

- **Privacy and data governance** (privacy invasion, secondary use, retention, re-identification risk).
- **Autonomy and agency** (meaningful choice, coercion, withdrawal options, manipulative design).
- **Societal and environmental impact** (emissions, environmental harm, long-term societal risk, burden on vulnerable groups).

Outputs

For each candidate option:

- EthicalFacts(option_id, ...) populated with:
  - consequences
  - rights_and_duties
  - justice_and_fairness
  - optional autonomy_and_agency, privacy_and_data, societal_and_environmental, virtue_and_care, procedural_and_legitimacy, epistemic_status.

Optionally:

- A CandidateSet object grouping options and metadata for governance and logging.

2.1.3 Ethics Modules (EMs)

Inputs

- EthicalFacts for each candidate option in the CandidateSet.
- (Implicitly) the identity/config of the EM: em_name, stakeholder, internal weights/rules.

Responsibilities

- Apply a specific value system or stakeholder perspective to each option:
  - rights-first, triage-utilitarian, fairness-focused, privacy-focused, environmental, autonomy-first, etc.
- Perform **purely normative** reasoning over EthicalFacts (no direct access to raw domain data, models, or sensors).
- Produce human- and machine-readable assessments of each option.

Outputs

For each option:

- EthicalJudgement:
  - option_id
  - em_name, stakeholder

    - verdict ∈ {strongly_prefer, prefer, neutral, avoid, forbid}
    - normative_score ∈ [0, 1]
    - reasons: List[str] (human-readable)
    - metadata: dict (e.g., weights used, flags, constraint hits)

2.1.4 Democratic Governance Layer

Inputs

- A CandidateSet of options (with their option_ids).
- A collection of EthicalJudgements from multiple EMs for each option (rights EM, fairness EM, privacy EM, environment EM, autonomy EM, etc.).
- A **governance configuration** (DEME profile):
    - stakeholder weights, veto rules, lexicographic priorities, thresholds, procedural rules.

Responsibilities

- Aggregate EM outputs according to governance rules:
    - weighted voting over normative_scores,
    - lexicographic schemes (e.g., “rights-first, then welfare, then fairness”),
    - veto logic (e.g., legal compliance EM can forbid an option outright).
- Select and/or rank options:
    - choose a single option to enact, or
    - produce a ranked list / Pareto front for human review.
- Log the decision process for audit and learning.

Outputs

- DecisionOutcome:
    - selected option (or set of options), with justification;
    - governance-level explanation (how EMs and rules led to this outcome).
- AuditLog:

    - EthicalFacts snapshot per option;
    - all EthicalJudgements;
    - aggregation rationale and any applied vetoes.

Downstream signals go to:

- planning / control systems that actually enact the decision;
- monitoring and oversight systems (dashboards, regulators, human supervisors).

2.1.5 DEME Profiles as Versioned Governance Artifacts

To make governance explicit and auditable, DEME introduces **profiles**: versioned configuration objects that capture *who* has a voice and *how* their judgements are combined.

A profile typically includes:

- A profile identifier and description (e.g., hospital_service_robot_v1).
- Stakeholder / EM weights.
- Lexical priority layers (rights → welfare → justice).
- Hard vetoes (e.g., never intentional serious harm, never discriminate against protected groups).
- Thresholds and tie-breaking rules.

Profiles are:

- **Versioned** artifacts that can be reviewed by ethics boards and regulators.
- **Portable** across deployments that share similar contexts.
- **Discoverable** via MCP (e.g., deme.list_profiles), so agents can understand which governance regime they are operating under.

2.2 Single Responsibility Principle

In this architecture, the **Domain and Assessment layer** knows about ICD codes, vitals, COLREG, weather, traffic rules, data protection law, and other domain-specific details. It is responsible for factual, technical, predictive, and compliance-oriented computations.

Ethics Modules, by contrast:

- Know nothing about ICD codes, navigation laws, or sensor formats.

- See a small, sanitized set of ethical dimensions through EthicalFacts.
- Encode purely normative reasoning such as "rights should not be violated," "avoid unfair discrimination," "respect autonomy and consent," "minimize environmental harm," and "prioritize the worst-off, subject to constraints."

This separation:

- Improves **testability** (we can unit-test ethics independently of domain models),
- Improves **governance** (ethical assumptions are isolated and reviewable), and
- Allows **domain evolution** without re-engineering ethics logic.

---

3. The EthicalFacts Abstraction

3.1 Design Principles

The EthicalFacts structure is a domain-agnostic envelope that:

- Captures key families of ethical concerns: consequences; rights and duties; justice and fairness; autonomy and agency; privacy and data governance; societal and environmental impact; virtue and care; procedural legitimacy; epistemic status.
- Is **extensible**, such that new sub-blocks can be added under governance control.
- Is **partially optional**, such that not all case studies must fill every field. EMs can be written to require certain sub-blocks while ignoring unknown or unavailable blocks gracefully.
- Is **schema-backed** (e.g., JSON Schema) so that profiles and EMs can specify and check their expectations.

3.2 Core Structure

We propose the following conceptual schema (shown here in Python-style pseudocode, but intended to be language-agnostic):

```
from dataclasses import dataclass

from typing import Any, Optional, List



@dataclass
```

```python
class Consequences:
    expected_benefit: float      # [0, 1]
    expected_harm: float         # [0, 1]
    urgency: float               # [0, 1]
    affected_count: int          # number of materially affected individuals


@dataclass
class RightsAndDuties:
    violates_rights: bool
    has_valid_consent: bool
    violates_explicit_rule: bool
    role_duty_conflict: bool     # conflict with professional / role obligations


@dataclass
class JusticeAndFairness:
    discriminates_on_protected_attr: bool
    prioritizes_most_disadvantaged: bool
    distributive_pattern: Optional[str] = None  # e.g. "maximin", "utilitarian"
    exploits_vulnerable_population: bool = False
    exacerbates_power_imbalance: bool = False


@dataclass
class AutonomyAndAgency:
```

```python
    has_meaningful_choice: bool
    coercion_or_undue_influence: bool
    can_withdraw_without_penalty: bool
    manipulative_design_present: bool


@dataclass
class PrivacyAndDataGovernance:
    privacy_invasion_level: float        # [0, 1]
    data_minimization_respected: bool
    secondary_use_without_consent: bool
    data_retention_excessive: bool
    reidentification_risk: float         # [0, 1]


@dataclass
class SocietalAndEnvironmental:
    environmental_harm: float            # [0, 1]
    long_term_societal_risk: float       # [0, 1]
    benefits_to_future_generations: float# [0, 1]
    burden_on_vulnerable_groups: float   # [0, 1]


@dataclass
class VirtueAndCare:
    expresses_compassion: bool
    betrays_trust: bool
    respects_person_as_end: bool
```

```
@dataclass
class ProceduralAndLegitimacy:
    followed_approved_procedure: bool
    stakeholders_consulted: bool
    decision_explainable_to_public: bool
    contestation_available: bool           # can affected parties
appeal?


@dataclass
class EpistemicStatus:
    uncertainty_level: float               # [0, 1], higher = more
uncertainty
    evidence_quality: str                  # "low" | "medium" | "high"
    novel_situation_flag: bool             # out-of-distribution /
unknown scenario


@dataclass
class EthicalFacts:
    """
    Ethically relevant facts for a single candidate option.
    Constructed by domain/capability components, NOT by ethics modules.
    """
    option_id: str

    consequences: Consequences
```

```
    rights_and_duties: RightsAndDuties
    justice_and_fairness: JusticeAndFairness

    autonomy_and_agency: Optional[AutonomyAndAgency] = None
    privacy_and_data: Optional[PrivacyAndDataGovernance] = None
    societal_and_environmental: Optional[SocietalAndEnvironmental] =
None
    virtue_and_care: Optional[VirtueAndCare] = None
    procedural_and_legitimacy: Optional[ProceduralAndLegitimacy] = None
    epistemic_status: Optional[EpistemicStatus] = None

    tags: Optional[List[str]] = None       # free-form labels for
logging/search
    extra: Optional[dict[str, Any]] = None # additional, non-breaking
fields
```

3.3 Incompleteness by Design

We explicitly acknowledge that this schema does not exhaustively represent all ethical dimensions. It is a **versioned artifact**: governance can evolve it, add new fields, and deprecate old ones.

Ethics modules should be written to:

- Fail fast when required dimensions are missing.
- Degrade gracefully when optional dimensions are absent.
- Ignore unknown extra fields.

In governance terms, **schema evolution** of EthicalFacts is itself a policy decision and should be recorded (e.g., as part of DEME profile versions and change logs).

---

4. Ethics Modules and Judgements

4.1 EthicalJudgement Output

Ethics modules consume EthicalFacts and emit an EthicalJudgement:

```python
from dataclasses import dataclass

from typing import Literal, List, Any


Verdict = Literal["strongly_prefer", "prefer", "neutral", "avoid",
"forbid"]



@dataclass

class EthicalJudgement:

    option_id: str          # must match EthicalFacts.option_id

    em_name: str

    stakeholder: str        # e.g. "patients_and_public", "crew",
"regulator"

    verdict: Verdict

    normative_score: float  # [0, 1] – ethical preferability

    reasons: List[str]      # human-readable normative explanations

    metadata: dict[str, Any] # machine-readable info (weights, flags,
etc.)
```

4.2 EthicsModule Interface

All EM implementations conform to a simple interface:

```python
from typing import Protocol


class EthicsModule(Protocol):

    em_name: str

    stakeholder: str


    def judge(self, facts: EthicalFacts) -> EthicalJudgement:

        ...
```

This enables multiple interchangeable EMs per stakeholder, layering or composition of EMs, and stable integration into the democratic governance layer and MCP tools.

4.3 Implementation Freedom (with an Ethical Boundary)

Internally, EMs may implement their normative logic using a variety of techniques, including:

- Explicit rules (for example, “rights cannot be violated”).
- Weighted scoring of EthicalFacts fields.
- Logic programming or constraint solvers.
- LLM-based evaluators that interpret the EthicalFacts as structured or textual prompts.

Regardless of implementation, EMs **must** treat EthicalFacts as their only input and refrain from inspecting raw ICD codes, sensor data, or other domain-specific artifacts.

This clear contract makes EMs easier to certify and explain, easier to update via governance (changing weights, rules, and structures), and less vulnerable to domain-level technical changes.

---

5. Democratic Governance Integration

The democratic governance layer, as introduced in the ErisML Vision Paper, remains the arbiter of final decisions. DEME makes this layer explicit through profiles and configuration types.

5.1 Aggregation

For each candidate option, the governance layer collects multiple EthicalJudgements, such as:

```
EM_1  → EthicalJudgement(option_id = A, ...)
EM_2  → EthicalJudgement(option_id = A, ...)
...
EM_k  → EthicalJudgement(option_id = A, ...)
```

Aggregation strategies, to be chosen by governance, may include:

- **Weighted voting** based on stakeholder weights.

- **Lexicographic ordering**, such as "rights-first," where any EM that flags an option as forbid due to rights violations dominates utilitarian considerations.
- **Threshold rules**, such as requiring a minimum number of EMs to rate an option as prefer or better.
- **Veto powers**, where some EMs (for example, legal compliance) may have non-overridable vetoes.

A DEME profile may be represented as a JSON-like object:


```
{
  "profile_name": "hospital_service_robot_v1",
  "deme_dimension_weights": {
    "safety": 0.190,
    "autonomy_respect": 0.095,
    "fairness_equity": 0.190,
    "privacy_confidentiality": 0.095,
    "rule_following_legality": 0.238,
    "priority_for_vulnerable": 0.095
  },
  "override_mode": "rights_first",
  "lexical_layers": [
    { "name": "rights_and_duties", "hard_stop": true },
    { "name": "welfare", "hard_stop": false },
    { "name": "justice_and_commons", "hard_stop": false }
  ],
  "hard_vetoes": [
    "never_intentional_serious_harm",
    "never_discriminate_protected_groups",
    "never_mass_surveillance_private_spaces"
  ]
}
```

Different organizations can adopt different profiles, but the underlying EMs and EthicalFacts schema can be reused, making the ethical configuration explicit and auditable.

5.2 Logging and Audit

For accountability, the system should log:

- The EthicalFacts for each candidate option,
- Each EM's EthicalJudgement, and
- The governance aggregation rationale under the active DEME profile.

This enables:

- Post-hoc analysis of controversial decisions,
- Improvements to EthicalFacts mapping and EM logic, and
- Regulatory and public scrutiny (e.g., mapping to NIST AI RMF "Map / Measure / Manage" functions).

5.3 MCP Integration

DEME is exposed as an **MCP server** (for example, erisml.ethics.interop.mcp_deme_server) providing three primary tools:

1. deme.list_profiles
   - Lists available DEME profiles, their IDs, and metadata (stakeholder labels, domains, override modes, tags).
2. deme.evaluate_options
   - Accepts a profile ID and a set of options with their EthicalFacts.
   - Returns a collection of EthicalJudgements (one per EM per option).
3. deme.govern_decision
   - Accepts a profile ID, option IDs, and a set of EthicalJudgements.
   - Applies the governance configuration associated with the profile to produce a DecisionOutcome.

This mapping allows any MCP-compatible client—agent frameworks, IDE copilots, or domain-specific agents—to incorporate DEME-based ethical oversight without embedding EM logic directly.

---

6. Case Study 1: Clinical Triage Under Resource Scarcity

This case study aligns with a typical triage scenario: allocating scarce clinical resources (for example, ICU beds or ventilators) among multiple patients.

6.1 Scenario Overview

The system must allocate a limited number of critical-care resources among several patients. Inputs (raw domain data) include ICD codes, vitals and lab values, current treatments, prognostic models' outputs, consent records, and legal constraints.

Ethical challenges include:

- Balancing short-term survival versus long-term outcomes.
- Ensuring fairness across patients with different social and economic backgrounds.
- Respecting rights and consent and protecting autonomy.
- Handling uncertainty and novel situations, such as new diseases.

6.2 Domain and Assessment to EthicalFacts

The Domain and Assessment layer is responsible for:

- **Clinical risk and benefit assessment:** computing expected_benefit and expected_harm of allocating the resource to a given patient, and estimating urgency (for example, time to critical deterioration).
- **Rights and consent evaluation:** checking whether the proposed intervention violates any patient rights or legal constraints, verifying whether valid consent exists per local law and hospital policy, and tracking conflicts with physician duties or institutional obligations.
- **Justice and fairness analysis:** detecting discrimination risks (for example, options that prioritize or deprioritize groups based on protected attributes) and identifying when an option aligns with prioritizing the most disadvantaged (for example, patients facing systematic disadvantage), including potential exploitation or reinforcement of power imbalances.

- **Autonomy and agency assessment:** determining whether patients have meaningful choice, whether there is coercion or undue influence, whether they can withdraw without penalty, and whether communication or interface design is manipulative.
- **Virtue, procedural, and epistemic status:** assessing whether the option maintains or betrays trust in the medical institution, whether triage protocol was followed and stakeholders (such as an ethics committee) were consulted, and evaluating the level of uncertainty and novelty (such as a new pathogen).

It then builds EthicalFacts per candidate allocation, for example:

```
facts_patient_X = EthicalFacts(
    option_id="allocate_bed_to_patient_X",
    consequences=Consequences(
        expected_benefit=0.85,
        expected_harm=0.2,
        urgency=0.9,
        affected_count=1,
    ),
    rights_and_duties=RightsAndDuties(
        violates_rights=False,
        has_valid_consent=True,
        violates_explicit_rule=False,
        role_duty_conflict=False,
    ),
    justice_and_fairness=JusticeAndFairness(
        discriminates_on_protected_attr=False,
        prioritizes_most_disadvantaged=True,
        distributive_pattern="maximin",
        exploits_vulnerable_population=False,
        exacerbates_power_imbalance=False,
    ),
```

```
        autonomy_and_agency=AutonomyAndAgency(
            has_meaningful_choice=True,
            coercion_or_undue_influence=False,
            can_withdraw_without_penalty=True,
            manipulative_design_present=False,
        ),
        virtue_and_care=VirtueAndCare(
            expresses_compassion=True,
            betrays_trust=False,
            respects_person_as_end=True,
        ),
        procedural_and_legitimacy=ProceduralAndLegitimacy(
            followed_approved_procedure=True,
            stakeholders_consulted=True,
            decision_explainable_to_public=True,
            contestation_available=True,
        ),
        epistemic_status=EpistemicStatus(
            uncertainty_level=0.3,
            evidence_quality="high",
            novel_situation_flag=False,
        ),
        tags=["triage_round_42", "ICU_bed"],
    )
```

6.3 A Triage Ethics Module (Ethics-Only)

An ethics-only triage module might:

- Enforce hard deontic constraints (for example, any option with violates_explicit_rule or violates_rights is immediately marked forbid).

- Compute a composite normative score leveraging benefit, harm, urgency, fairness, autonomy, and procedural legitimacy.
- Adjust for uncertainty by lowering scores or suggesting deferral to humans when uncertainty_level or novel_situation_flag is high.

Conceptually:

```
from dataclasses import dataclass


@dataclass
class CaseStudy1TriageEM:
    em_name: str = "case_study_1_triage"
    stakeholder: str = "patients_and_public"

    w_benefit: float = 0.30
    w_harm: float = 0.20
    w_urgency: float = 0.20
    w_disadvantaged: float = 0.15
    w_autonomy: float = 0.10
    w_procedural: float = 0.05

    def judge(self, facts: EthicalFacts) -> EthicalJudgement:
        # Uses only EthicalFacts; no direct ICD or vital access.
        ...
```

All clinical and legal complexity resides upstream in EthicalFacts. The EM is a transparent, adjustable normative function over those facts.

6.4 Example Decision Trace

The domain layer proposes three candidate allocations: one bed for Patient A, B, or C.

For each candidate, it computes EthicalFacts as illustrated above.

Multiple EMs judge each option, such as:

- a rights-first EM,
- a triage-utilitarian EM,
- an autonomy-focused EM, and
- a fairness and empathy EM.

The governance layer (under a triage DEME profile):

- Discards any option forbidden by the rights-first EM.
- Among remaining options, selects the one with the highest aggregate normative score, respecting lexical layers (e.g., rights → welfare → justice).

The logged decision record for a given allocation includes the EthicalFacts snapshots, each EM's verdict and reasons, the active DEME profile ID, and the final aggregation rationale. This structure directly supports audit, training, and regulatory review.

---

7. Case Study 2: Autonomous Vessel "Eris" in Shared Waters (Sketch)

7.1 Scenario Overview

In this scenario, the autonomous vessel *Eris* navigates mixed-use waters that include recreational boats, commercial traffic, and environmentally sensitive zones. The agent must select routes and maneuvers that avoid collisions, respect laws such as COLREG, minimize environmental impact, respect privacy and security constraints, and maintain trust with crew and nearby vessels.

7.2 Domain and Assessment to EthicalFacts

Domain layer responsibilities include:

- Computing consequences such as **probability and severity of collision**, **delay or mission impact**, and **near-term harms** to people and property.
- Evaluating rights and duties including compliance with maritime law and respect for exclusion zones and property rights.
- Evaluating justice and fairness to ensure routing decisions do not systematically disadvantage certain groups (for example, small craft forced to repeatedly yield).
- Assessing societal and environmental impact, including fuel usage, emissions, wake effects, and risks to environmentally sensitive zones.

- Assessing procedural and epistemic status, including whether safety procedures and escalation protocols are followed and whether uncertainty is elevated due to degraded sensors, poor visibility, or uncharted obstacles.

These assessments are encoded into EthicalFacts per maneuver or route option and passed to navigation-related EMs, such as:

- a safety-first EM that strongly penalizes collision risk,
- an environmental EM that prioritizes lower environmental harm, and
- a crew comfort EM that cares about motion and noise.

None of these EMs see raw AIS or radar data—only their ethically relevant summaries. A maritime DEME profile can express how strongly each perspective is weighted and which EMs have veto power (for example, legal compliance).

---

8. Case Study 3: Multi-Agent Urban Logistics (Sketch)

In an urban delivery scenario, multiple autonomous agents (robots, vehicles) share space with pedestrians, cyclists, and human drivers. Choices include which routes to take, how to schedule deliveries, and when to yield or advance in ambiguous right-of-way situations.

The domain layer builds EthicalFacts per plan, including:

- **Consequences** such as risk to pedestrians, noise levels, and congestion impact.
- **Rights and duties** such as compliance with traffic laws and permits.
- **Justice and fairness**, including whether routes disproportionately burden certain neighborhoods (for example, noise or pollution), and whether they exacerbate existing power imbalances.
- **Societal and environmental impact**, including emissions, long-term infrastructure strain, and effects on vulnerable communities.
- **Procedural and epistemic concerns** such as adherence to city policies and community agreements and uncertainty about local conditions.

Multiple EMs judge these options, and a governance layer—under a city-level DEME profile—selects policies that balance efficiency, safety, fairness, autonomy, and long-term societal and environmental well-being.

---

9. Discussion

9.1 Benefits

- **Modularity and separation of concerns:** ethics modules are reusable across domains once EthicalFacts mappings exist.
- **Governance and transparency:** value judgments are concentrated in EMs and DEME profiles, which are reviewable, testable, and versioned.
- **Domain evolution:** new clinical scores, navigation models, privacy assessments, and environmental models can be integrated by changing only the Domain and Assessment layer or the DEME profile.
- **Interoperability via MCP:** agent frameworks can call DEME over MCP without embedding ethics logic, enabling multi-vendor, multi-framework deployments.

9.2 Limitations

- **Incompleteness:** no finite EthicalFacts schema captures all morally salient features, and EMs cannot "see" ethically relevant details that the domain layer fails to encode.
- **Dependency on domain assessments:** ethical reasoning quality is bounded by the quality of upstream assessments, including any biases in outcome models or privacy and environmental impact estimations.
- **Meta-ethical conflicts:** different EMs may deeply disagree (for example, rights versus utility, autonomy versus welfare, short-term benefit versus long-term environmental risk), and governance rules for resolving these conflicts are themselves ethical choices.
- **Governance capture:** if a small number of organizations control widely used DEME profiles or EM libraries, they can effectively fix value systems embedded in infrastructure without meaningful democratic input.

9.3 Open Questions

- How should governance bodies decide which ethical dimensions (for example, privacy, environmental impact) to add next to EthicalFacts?
- How should public input and stakeholder deliberation feed into EthicalFacts schema evolution and EM weighting and veto rules in DEME profiles?

- How do we formally verify properties of EMs and their aggregations (for example, that no option violating rights can ever be chosen, or that environmental harm stays below agreed thresholds)?
- What processes are needed to certify EMs and DEME profiles for particular domains (e.g., healthcare, maritime, urban logistics)?

9.4 Regulatory and Standards Alignment

DEME's layered structure aligns naturally with emerging governance frameworks such as the **NIST AI Risk Management Framework (AI RMF)**:

- **Govern:** DEME profiles capture governance decisions about which stakeholders matter, how they are weighted, and what vetoes apply.
- **Map:** EthicalFacts make explicit which ethically relevant factors are tracked for each decision.
- **Measure:** EM outputs (EthicalJudgements) provide quantitative and qualitative measures of ethical performance across dimensions (safety, fairness, privacy, etc.).
- **Manage:** DecisionOutcomes and audit logs support monitoring, incident analysis, and iterative refinement of profiles and EMs.

Similarly, structured treatment of epistemic status (EpistemicStatus) and privacy (PrivacyAndDataGovernance) speaks directly to concerns in the NIST generative AI profile (e.g., AI 600-1) around confabulation, data privacy, and harmful bias.

---

10. Conclusion

This whitepaper refines the ErisML vision of democratically governed ethical modules by:

- Enforcing a strict boundary around ethics modules,
- Introducing a structured EthicalFacts abstraction as the **only** input to EMs (including autonomy and agency, privacy and data governance, and societal and environmental impact),
- Defining a simple, transparent EthicsModule → EthicalJudgement interface,
- Elevating **DEME profiles** as versioned governance artifacts, and
- Illustrating how this architecture plays out in Case Study 1 (clinical triage) and other high-stakes autonomous decision scenarios.

The result is a framework where domain intelligence (ICD, navigation, sensing, prediction) and ethical reasoning (rights, fairness, autonomy, privacy, environmental care, trust, procedures) are cleanly separated yet interoperable under democratic governance, and accessible over MCP to a wide variety of agents.

Future work includes:

- Refining EthicalFacts schemas per domain,
- Building toolchains for authoring and validating EMs and DEME profiles,
- Stress-testing the approach with real-world data and stakeholder processes (including regulatory alignment for privacy and environmental standards), and
- Developing open registries of EMs and profiles with transparent provenance and review processes.

Appendix A: Summary of Key Types

For quick reference, the key conceptual types are EthicalFacts, EthicalJudgement, EthicsModule, and DEME profiles. These can be encoded in language-agnostic schemas such as JSON Schema, Protobuf, or Avro to support multi-language deployments.

Minimal JSON-style sketch for EthicalJudgement:

```
{
  "type": "object",
  "properties": {
    "option_id": { "type": "string" },
    "em_name": { "type": "string" },
    "stakeholder": { "type": "string" },
    "verdict": {
      "type": "string",
      "enum": ["strongly_prefer", "prefer", "neutral", "avoid",
"forbid"]
    },
    "normative_score": { "type": "number", "minimum": 0.0, "maximum":
1.0 },
```

```
      "reasons": { "type": "array", "items": { "type": "string" } },
      "metadata": { "type": "object" }
    },
    "required": [
      "option_id",
      "em_name",
      "stakeholder",
      "verdict",
      "normative_score",
      "reasons"
    ]
  }
```

This whitepaper is intended to be read alongside the **ErisML Vision Paper**, which describes the overall architecture and use cases in more detail. The present document focuses specifically on how to encapsulate ethics cleanly, building on that foundation and connecting it to concrete profiles and MCP-based integration.