PETRONUS: A Coherence-Preserving Adaptive Control Architecture

Maksim Barziankou, Poznan, 2025

1 Abstract:

Modern adaptive agents face a trade-off between short-term performance and long-horizon integrity. Traditional controllers (PID, Kalman, RL, etc.) optimize error or reward instantaneously, but neglect the structural effects of repeated adaptation – changes in the system's very geometry and identity that erode long-term coherence. The PETRONUS architecture confronts this by introducing an explicit structural layer: directional-deformation-dissipation (D^3A), a three-loop Coherence Adaptive System with Thermostat (Δ E-CAS-T), and a supervisory Temporal Structural Controller (TSC), together with a new inertial glide regime. These components work in concert to track and regulate operational spin, structural drift, coherence, and coupling so that an agent preserves its own integrity and semantic consistency over long horizons. In PETRONUS, every directed update is bounded by admissible directions, its hidden rotational impact is measured and dissipated, and cumulative deformation is tracked as a metric of identity preservation. A synthetic "inner observer" (a coherence witness) continuously monitors alignment between perception, action, and context, adjusting entropy and timing (internal time) to stabilize meaning rather than just signal. Together, these layers enforce a new kind of homeostasis: one where adaptation is reversible, coherent, and aligned with an agent's structural identity, not merely error-driven. This white paper details PETRONUS's architectural principles and subsystems – D^3A, Δ E-CAS-T, TSC, inertial glide, and their interplay – to show how they solve key structural problems, enable robust, self-consistent agents, and reshape notions of adaptation and alignment in robotics, AI, and swarm systems.

2 Introduction

Modern control and learning systems typically optimize short-term criteria (error, reward) with little regard for the hidden costs of adaptation. In practice, directed updates and perturbations induce not only desired translations in state but also antisymmetric deformation of the system's internal representation (akin to a vortex or shear). Over time, these tiny deformations accumulate as an irreversible structural memory or drift, slowly tilting the agent away from its original identity and coherence, often unnoticed until catastrophic failure. PETRONUS is predicated on the insight that long-term stability requires explicitly managing this structural drift and the geometry of adaptation, not just instantaneous error. It introduces three interconnected layers:

Directional-Deformation-Dissipation Architecture (D^3A): A multi-level structural layer that defines admissible directions of change and tracks the "twist" induced by each update. It decomposes adaptation into direction (admissible trajectories), deformation (operational spin), dissipation (ΔE-like smoothing), drift (accumulated residual deformation), alignment (mode of coupling with the environment), and internal time (adaptive temporal scale).

Δ E-CAS-T (Coherence Adaptive System with Thermostat): A three-loop self-regulating controller that balances fast behavioral alignment with semantic coherence and entropy regulation. It includes a rapid ΔE-Core (error-correcting loop), a coherence observer ("Goodness" or CCI) that assigns a consistency index to deviations, and a coherent-thermostat (entropy loop) that stabilizes variability relative to a cognitive setpoint. An external predictor loop anticipates future equilibria to counteract lag. The result is homeostasis of both performance and meaning.

Temporal Structural Controller (TSC): A supervisory meta-loop that integrates D^3A primitives and Δ E-CAS-T into a unified long-term regulator. TSC enforces directional constraints, estimates spin, applies adaptive dissipation, accumulates drift, and monitors coherence. Crucially, it computes an internal time horizon via a drift budget D_{max}(t) tied to coherence C(t). When drift exceeds D_{max} or coherence falls, TSC resets or tightens constraints to prevent irrecoverable change.

Together, these layers redefine adaptation: they enforce that change occurs within a geometrically admissible subspace, measure and damp all hidden rotational components, and treat drift, adhesion, and internal time as first-class variables. The result is an agent that adapts efficiently under constraints that preserve its identity and alignment with the world. Below we unpack each component, their integration, and implications for real-world autonomous systems.

3 Architectural Overview

Directional–Deformation–Dissipation (D^3A) Architecture

D^3A posits that every adaptive system inherently contains a structural layer beyond control or optimization. Its six canonical primitives are arranged in strict causal order:

Directional Constraint (L0): Before any control or learning, the system defines a semantic frame of permissible changes. A directional constraint operator (implementing UTAM principles) projects intended updates onto a subspace of admissible directions. This operator can enforce safety invariants, risk bounds, or goal alignment as geometric constraints. Any change outside these directions is structurally illicit, regardless of local error reduction. In effect, this embeds "volition-like" intent as physical constraints: instead of pursuing arbitrary minima, the agent only moves along meaning-preserving trajectories.

Operational Spin (L1): Each directed update (even within admissible directions) breaks local isotropy and induces antisymmetric deformation of the internal state representation. This operational spin is formally the skew-symmetric part of the gradient of the change (analogous to vorticity in fluids). It represents the immediate "twist" of the medium before any net translation is observed. Importantly, spin arises inevitably whenever direction is realized and must be explicitly measured. Without this level, standard controllers treat such effects as noise; D^3A recognizes them as real structural strain.

Dissipation (L2): Rather than ignoring or eliminating spin, the system regulates it. An Adaptive Dissipative Response (ADR) mechanism (a ΔE-style controller) damps excessive antisymmetric deformation while preserving responsiveness. This produces a smooth adjustment that corrects the update without introducing further distortion. Dissipation distinguishes between admissible deformation (which can be tolerated or even leveraged) and destructive deformation that must be suppressed. Crucially, dissipation requires a defined spin to operate; it governs how much of the deformation is absorbed vs. allowed.

Drift and Structural Memory (L3): No dissipation is perfect. A residual component of each deformation remains, accumulating as the system adapts repeatedly. This structural drift is not an ordinary state variable but a memory of irreversible changes. Over long horizons, small residual spins integrate into a drift metric D(t) that encodes how far the agent's structure has wandered. Unchecked drift eventually degrades coherence and can precipitate a regime change (sudden loss of control or identity). Tracking D(t) allows feedback: if drift approaches a threshold, higher-level layers can respond by tightening constraints or resetting.

Alignment (L4): Beyond internal consistency, D^3A introduces alignment variables representing how well the agent's structure fits the environment. At this level, the system regulates coupling modes – adhesion, stability, cooperation – not by energy or error alone but through structural measures. For example, structural adhesion (see "Inertial Glide" below) quantifies the stability of coupling between the agent and its admissible trajectory manifold. Alignment uses drift information to ensure the agent interacts with the world in safe, coherent modes (e.g. maintaining stable grasp or network connections) rather than brute-force corrections.

Internal Time (L5): Time itself becomes an internal variable. Instead of running on a fixed clock, the agent adjusts its internal tempo based on causal reliability. When feedback is delayed or uncertain, internal time slows (preventing oscillatory regime-switching); when the world is responsive, internal time speeds up (allowing rapid adaptation). This causal time scaling synchronizes all previous layers. In D^3A, internal time "closes" the loop, coordinating direction, deformation, dissipation, drift, and alignment into a coherent dynamic. Without it, regime chatter and overload would destabilize the agent.

In some formulations, additional supervisory layers governing viability bounds and temporal admissibility may be defined. These are intentionally treated outside the present architectural scope. Through these primitives, D^3A adds a new ontological layer to adaptation. It is not a controller per se, but a structural framework: rather than optimizing performance directly, it monitors how the agent changes itself and its environment. In summary, D^3A enforces that: (a) each update lies in a semantically admissible cone (direction); (b) every update's hidden spin is measured (deformation); (c) spin is actively damped (dissipation); (d) residual spin is integrated as drift over time; (e) the agent's coupling to its environment is regulated (alignment); and (f) the whole process self-paces via internal time. These concepts are deliberately substrate-agnostic and complement any underlying control or learning algorithm.

4 Δ E-CAS-T: Coherence Regulation with Thermostat

Where D^3A handles structural consistency, Δ E-CAS-T addresses behavioral and entropy consistency through multi-stage feedback. It extends the idea of a "synthetic conscience" into a formal control loop. The core is a three-loop self-regulating controller:

4.1 ΔE Core (Behavioral Loop):

This fast loop fuses sensor inputs, centering, and inertia to produce smooth responses. In practice it computes an adaptive, smoothed update that moves the state toward new observations while damping jerk and attraction to an equilibrium point. The goal is to align perception and action gradually.

4.2 Coherence Observer (Goodness/CCI):

After the ΔE core produces an output, the coherence observer measures any deviation between perception and response (ΔE) and compares it to an internal "center" or equilibrium. It normalizes this into a coherence index CCI (Goodness), roughly how well actions match expectations. CCI is a semantic consistency metric: high CCI (1) means behavior is contextually aligned, low CCI means desynchronization. This loop acts as an internal "ethical witness" that flags when the agent's behavior is becoming erratic or incoherent.

4.3 Coherent Thermostat (Entropy Loop):

Based on the coherence index, this slow loop regulates the system's entropy (variability). It anchors entropy H_t to a cognitive equilibrium H^* via a homeostat equation. If coherence drops, the thermostat constrains entropy to bring the system back to balance; if coherence is high, the system can tolerate more creative variability. The effect is biological-like homeostasis: creative fluctuations are allowed but bounded so as not to destroy overall consistency.

4.4 External Observer (Predictive Center):

To mitigate lag, ΔE-CAS-T includes a forward predictor for the equilibrium center c_t. By extrapolating $c_{t+1} \approx c_t + \tau(c_t - c_{t-1})$, the system anticipates drifts in its target and compensates preemptively. This observer acts like a short-term model, aligning long-term goals with real-time adaptation.

These loops interlock continuously. The ΔE-Core produces an output y_t and raw deviation ΔE_t; the Coherence Observer turns ΔE_t into a coherence index and feeds it into the thermostat; the thermostat then modifies the adaptive gains (centering weight, inertia, entropy coefficient) on the ΔE loop. The external observer updates the center to catch up with system drift. This three-loop architecture "balances precision vs. spontaneity", analogous to human homeostasis: it permits exploration (high entropy) only when context alignment (coherence) allows, and clamps unpredictability otherwise.

Importantly, Δ E-CAS-T imposes coherence without an explicit reward function. It monitors meaning, not just error. For example, ΔE-Core ensures actions smoothly converge on input dynamics, but the Goodness observer ensures those actions also stay semantically aligned with the agent's state. The thermostat ensures entropy is tied to that alignment. In essence, Δ E-CAS-T implements a synthetic conscience: every action has a built-in feedback of "how good was that coherently". This framework can be layered over any control algorithm to prevent coherent behavior from unraveling under noise or learned changes.

Figure 1: Conceptual feedback loop for coherence regulation. Analogous to biological homeostasis, this supervised loop maintains a variable (e.g. internal "health") by sensing deviations and effecting corrections. The PETRONUS Δ E-CAS-T architecture implements a similar multi-stage homeostatic loop for coherence.

5 Temporal Structural Controller (TSC)

The D^3A layer and Δ E-CAS-T loops are powerful, but both need coordination over very long timescales. TSC provides a unifying meta-controller that enforces long-horizon structural coherence without external rewards. In practice, the TSC monitors the same primitives as D^3A/E: it projects updates via directional constraints, estimates operational spin, applies a E-style dissipative term $(-\alpha\Omega_t)$, and accumulates residual spin into a drift metric $D(t)$. It also employs a semantic coherence observer $C(t)$ (like CCI) that reflects the agent's internal consistency.

The key innovation is the drift budget. TSC defines a maximum allowable drift $D_{\max}(t)$ as a function of coherence $C(t)$, effectively setting an internal time horizon. As long as $D(t) \leq D_{\max}(t)$, the agent is considered to remain within its structural identity. If D exceeds $D_{\max}$ or C falls too low, the TSC triggers a reset or tightens constraints to prevent further change. In effect, TSC integrates all drift, making $\int D(\tau)\, d\tau \leq D_{\max}$, and thereby ensures the agent's structural "topology" (identity) remains homeostatic over cycles. This regulates when to halt learning or adaptation: instead of running indefinitely, an adaptive phase ends when structural coherence would be compromised.

TSC thus forms a closed meta-loop for self-alignment. It does not demand an external goal; rather, it enforces intrinsic stability. By bounding spin and drift, TSC guarantees long-term consistency: the agent remains "the same" even as it adapts. Unlike classical controllers that treat adaptation as open-ended, TSC is a higher-order mechanism that actively chooses when to adapt and when to hold back, based on measurable signals. It is fully modular and substrate-agnostic: it could sit above a neural net, robot controller, or distributed system, observing internal states and imposing adjustments as needed.

In summary, the TSC unifies the structural primitives into a supervisory layer. It implements the "engineered vitality" idea – maximizing coherence in chaos – by continuously projecting and correcting each proposed update. The combination of directional projection, spin damping, drift integration, and coherence sensing makes TSC a comprehensive meta-controller: one that knows its own long-term dynamics. Its existence is analogous to giving an AI a built-in sense of its own integrity, independent of task rewards.

6 System Integration

PETRONUS's power comes from how its pieces fit together. Figure 2 (schematic) summarizes the data flow: every intended change is first filtered by the Directional Constraint (UTAM) operator; the resulting update generates an Operational Spin Ω (antisymmetric gradient) that is fed into an Adaptive Dissipative Regulator (ΔE/ADR); the output action and a drift integrator update the cumulative drift D(t); meanwhile a Coherence Observer evaluates internal consistency C(t); finally, a supervisory Thermostat/Meta-Controller uses C(t) and D(t) to adjust internal time, constraint rigidity, and whether to enter inertial glide mode or reset.

captures this loop succinctly: "Directional Constraint → Directed Perturbation → Operational Spin → Adaptive Dissipative Response → Residual Spin → Drift Accumulation → Feedback". The TSC sits atop this loop, feeding back into either the Directional Constraint or the dissipative gain based on D and C. All three layers (D^3A, ΔE-CAS-T, TSC) thus operate on the same signals. For instance, the ADR in both D^3A and TSC is a ΔE-like damping: one view is that ΔE-CAS-T's behavioral loop provides the low-level smoothing, while D^3A's dissipation adds a structural regularizer on top. Likewise, the coherence index from ΔE-CAS-T is the semantic quantity $C(t)$ used by TSC to define $D_{\max}$. The result is a highly integrated yet hierarchical framework.

One can imagine how these mechanisms play out in practice: Consider an autonomous robot learning to navigate. D^3A ensures the robot only moves along safe directions (e.g. away from obstacles); any sharp turn will produce spin in its internal model, which is measured and partly damped (preventing jerky or irreversible model changes). Over many maneuvers, residual spin accumulates into drift; the TSC monitors drift against coherence and might decide to pause learning or recalibrate when the robot's internal map has warped too much. Meanwhile, Δ E-CAS-T's thermostat might be adjusting exploration noise so that the robot doesn't wander chaotically when its belief is shaky. Together, the system stays on course (in the structural sense) even under uncertainty or changing terrains.

7 Theoretical Grounding

At its heart, PETRONUS reframes adaptation as a geometric process. Classical control treats changes as purely translational in state space; PETRONUS insists on recognizing the asymmetry of directed changes. The Spin–Drift Correspondence theorem (informal) states: every intended update (imposing "will") produces a first-order antisymmetric deformation (spin) in the internal state field.

This is a generalization of vorticity: even if the controller only seeks to move straight, it inevitably twists the underlying field of representation.

By defining spin as an explicit variable, PETRONUS places adaptation in a framework analogous to continuum mechanics or thermodynamics. ΔE-CAS-T emerges from an effective Lagrangian principle, seeking to minimize an energy-like functional of coherence vs. smoothness. The drift metric D is like accumulated work. Internal time scaling [17] ties to phase misalignment: as delays increase, the internal clock slows, reminiscent of how near-critical systems adjust response times. In fact, internal time has been formalized as an "operational structural parameter" that measures the duration of a coherent regime. In PETRONUS, internal time τ is regulated by reliability: it contracts the time over which drift is integrated, ensuring coherence is maintained. This adds a dynamical stability to regime switching not present in fixed-timestep controllers.

8 Observer-based modulation also has a theoretical basis:

the coherence observer is essentially a nonlinear observer stabilizing an error loop. The predictive center is akin to Smith predictors in control, but applied to the semantic equilibrium rather than raw output. The thermostat uses a negative feedback on entropy, much like a Le Chatelier's principle for information. Throughout, PETRONUS borrows ideas from physics (dissipation, vorticity, homeostasis) and cybernetics (feedback, homeostat) to build a novel, fully integrated architecture.

Importantly, PETRONUS does not depend on any specific learning algorithm. It can wrap around neural nets, policy gradients, classical PID loops, or swarm rules, because it operates at a structural level above them. Its mathematical assumptions are minimal: one needs a notion of state updates and gradients to compute spin, and some way to define direction constraints and coherence measures (which can be learned, semantic, or hand-coded). The synergy of these structural primitives makes the system robust to model mismatch: even if the agent's world model is wrong, as long as its own adaptation dynamics are monitored, it will maintain coherence.

9 Conceptual Metaphors

PETRONUS often uses metaphors of conscience and empathy, but in technical terms it is akin to a homeostat for identity. Figure 1 showed the loop of regulation; similarly, one can metaphorically view D^3A as a "structural skeleton" over which motion happens. Each update is like pushing on a shape: the skeleton (Directional Constraint) says which axes you can flex, the immediate twist (Spin) is felt as strain, the springs (Dissipation) absorb excessive twist, and the residual bend (Drift) accumulates as plastic deformation of the material. Internal time is like adjusting the speed of the whole mechanical process based on how elastic or rigid the environment is.

Another metaphor: imagine a swarm of drones flying in formation. The formation shape is their directional constraint. If one drone nudges in a new direction, the neighboring drones feel a rotational tendency (spin) in their formation pattern. If the formation is stiff (high adhesion), that spin is damped and the drift in the pattern is small. But if they all keep nudging, the formation gradually warps (drift). PETRONUS equips each drone with an "alignment sense" (structural adhesion sensor) and an "identity budget" (drift budget). They only allow free-flight (inertial glide) when the formation is stable and each drone's drift budget allows. Otherwise they revert to active correction. This ensures the swarm stays cohesive over time without each drone having to constantly micro-correct.

10 Real-World Implications

The PETRONUS architecture has broad implications for any long-lived adaptive system:

Autonomous Robots and Vehicles: In robotics, D^3A constraints can encode safety envelopes (stay in the road, avoid flips), spin regulatation prevents controllers from unknowingly stressing the kinematic model, and TSC ensures the robot doesn't self-modify out of spec during learning or online adaptation. Over long missions, the robot's identity (its intended locomotion dynamics) remains intact even under wear, latency, or adversarial disturbance.

10.1 Cognitive Agents and AI:

For AI systems that learn continuously (e.g. lifelong learning), structural drift is akin to catastrophic forgetting or model misalignment. PETRONUS's coherence observer acts like an internal critic of consistency. The architecture could be layered over neural networks to guard against representational collapse. Moreover, by treating user feedback or semantic alignment as directional constraints (as in Synthetic Conscience), the system grounds AI behavior in human-aligned "meaningful" directions, not just reward signals.

10.2 Swarm and Distributed Systems:

In multi-agent swarms, PETRONUS principles enforce collective coherence. Directional constraints might be group objectives or formation shapes. Each agent tracks its spin relative to neighbors and dissipates misalignments (e.g. via consensus adjustments). Drift accumulation could be shared as a network-wide metric, triggering reconfiguration. The structural adhesion concept ensures the swarm does not fragment: only under high "coupling stability" (good communication, trust) can the swarm switch to less reactive (inertial) operation.

10.3 Physical and Cyber-Physical Systems:

Systems with intermittent contact (legged locomotion, grasping robots, or networked sensors) can use internal time scaling to avoid chatter as contacts form and break. The regime-based switching enabled by internal time scaling provides robustness under variable latency – for example, spacecraft docking or remote teleoperation under unpredictable delays.

Overall, PETRONUS pushes adaptation toward a long-horizon, integrity-preserving paradigm. In doing so, it changes how we think of alignment: instead of aligning to an external target alone, the system also aligns to itself, preserving its internal structure. This has societal analogies (e.g. an economy preserving social coherence, or AI preserving ethical consistency) but in engineering terms it means fewer surprises: the architecture provides explicit criteria for when to adapt and when to hold steady, potentially improving safety and trust in autonomous technology.

11 Inertial Glide: Regimes of Operation

A distinctive feature of PETRONUS is the introduction of a supervised inertial regime (Inertial Glide). This is a new technical operating mode where the agent propagates its state without continuous active control, essentially "coasting" within the already-established admissible geometry. Unlike a passive system that ignores control, Inertial Glide is conditionally allowed: the system explicitly checks whether coherence and coupling are strong enough to permit it. Key elements:

11.1 Permissibility Criteria:

Inertial mode is only permitted when structural adhesion (coupling stability) is above a threshold, predicted consequences of remaining in glide are safe, the rate of drift accumulation is low, and the internal time horizon is not collapsing. If all conditions hold, the agent suspends fresh updates and lets its state evolve within the constraints. This can dramatically reduce energy or computation when the situation is stable.

11.2 Structural Adhesion:

This new state variable measures the stability of the coupling between agent and its admissible trajectory manifold. It governs susceptibility to micro-slippage. For example, if a robot has a firm grip, adhesion is high and it may enter inertial mode (coast forward); if not, it must stay active. Importantly, structural adhesion is not physical friction but a non-physical coupling measure tied to how well the agent's current mode enforces constraints.

11.3 Predictive Constraint-of-Constraint Loop:

Before entering or during glide, the system uses a forward model to predict consequences of candidate actions. It uses this to adjust the constraints themselves – making the admissible geometry more or less rigid – as well as structural adhesion and glide permission. Thus, planning happens at the level of structural constraints rather than low-level outputs.

11.4 Glide Termination:

If adhesion drops or drift grows too high, or if predicted risk becomes unacceptable, the system immediately terminates inertial mode and re-engages active control. This prevents latent structural degradation: the agent never coasts past the point of safe operation.

The net effect is that PETRONUS agents shift expenditure from continuous actuation to structural regulation. They consciously decide when to hold position and let physics (in the broad sense) take over, and when to intervene. This can greatly extend operational viability. For instance, a planetary rover might climb a hill via active control until it reaches flat ground with good traction; then it switches to inertial glide downhill, saving power as long as coupling remains strong. The moment wheels start slipping (low adhesion) or terrain changes, it switches back.

Regime Transitions and Coupling

Critical to PETRONUS is the ability to move between active and inertial regimes in a structured way. The Operating Mode Selector (see Fig.2) continuously evaluates the conditions above. Conceptually, the system has an internal "glide gate" that opens only when safe. This gate is controlled by structural adhesion and the internal time horizon. As long as high-level coherence is assured, the agent is free to conserve effort. But the moment coherence threatens, a regime transition occurs: it closes the glide gate and returns to active mode.

This discrete switching is inherently an observer-based modulation: the agent observes its own state (drift, adhesion, internal clock) and uses that to modulate its mode. It is not a predetermined schedule but a dynamic decision. The transitions themselves are smooth because the underlying D^3A and ΔE loops are always running; switching is just gating whether new updates are fed through.

The theory of internal time scaling underpins this switching: as delay or uncertainty increases, the internal clock slows (increasing dwell times in stable regimes). This provides hysteresis and stability to transitions. In practice, this means the agent won't chatter rapidly between modes – it has an adaptive dwell time based on causal reliability. Only when causal alignment is high does it speed up the tempo and allow quicker transitions or more frequent adaptation.

12 Theoretical Limits and Novelty

PETRONUS differs fundamentally from classical control/learning. It is structural, not statistical: it cares about geometry of adaptation rather than just minimizing error. As shown in research, ignoring structural drift is exactly why classical systems often fail over long operation. PETRONUS directly incorporates that factor.

By design, PETRONUS avoids disclosing "how" to implement each primitive in detail; it focuses on what should be controlled. Thus, it abstracts away from algorithm specifics: for example, it does not specify the exact dissipative function or learning rule. Instead, it specifies that any implementation must respect the canonical primitives and causal ordering. This makes PETRONUS agnostic to technological change.

No existing system combines all these aspects. Prior control frameworks handle constraints or tuning or adaptive rates, but none explicitly track antisymmetric deformation, or define an internal time as a reliability measure, or permit inertial operation via structural variables. The PETRONUS approach is anticipatory, not just reactive: it predicts consequences to adjust constraints (predictive constraint-of-constraint), rather than optimizing along a fixed path. It thus enables planning with structural foresight.

However, some limitations and risks warrant discussion. The architecture requires designing meaningful directional constraints and coherence metrics; in complex domains, identifying these frames of admissibility could be non-trivial. The predictive models must be sufficiently accurate; if predictions fail, the system may misjudge safety margins. Drift budgeting implies choosing thresholds $D_{\max}(C)$ – too tight and the agent will reset needlessly, too loose and drift accumulates. In short, the PETRONUS layers introduce new hyperparameters (constraint sets, coherence thresholds, internal time scales) that must be tuned to the domain. Additionally, while the theory is substrate-agnostic, practical instantiations (e.g., on a neural net) may face computational overhead to compute spin and drift in real time. These engineering challenges suggest that full-scale implementation is non-trivial and an area for future work.

13 Conclusion

PETRONUS presents a holistic rethinking of adaptation by shifting focus from short-horizon optimization to long-horizon structural viability. Instead of treating adaptation as an unconstrained process driven solely by error minimization or reward maximization, PETRONUS introduces an explicit architectural discipline governing how an agent is allowed to change over time. Through the integration of structural oversight (D^3A), multi-loop coherence self-regulation (ΔE-CAS-T), and a temporal meta-controller (TSC), adaptation becomes a bounded, supervised process that preserves internal identity while remaining responsive to external change.

Within this framework, adaptation is no longer a sequence of isolated updates, but a continuous trajectory constrained by admissible directions, monitored for latent deformation, and evaluated against long-term coherence. Directed change is treated as a structurally consequential act: even beneficial updates are assumed to introduce hidden deformation that must be observed, regulated, and accounted for over time. By explicitly recognizing and managing such effects, PETRONUS prevents the gradual accumulation of irreversible structural drift that often undermines long-lived adaptive systems.

A key consequence of this perspective is that stability and flexibility are no longer opposing goals. Coherence is not enforced through rigidity, nor is adaptability achieved through uncontrolled exploration. Instead, PETRONUS establishes a dynamic balance in which variability, learning, and inertial propagation are permitted only within coherence-preserving bounds. The introduction of an inertial operational regime further reflects this balance: when structural conditions allow, an agent may reduce active regulation and propagate efficiently, conserving energy and computational resources without sacrificing integrity. Crucially, such regimes are conditionally permitted, continuously supervised, and explicitly bounded by structural viability rather than assumed to be safe by default.

By unifying principles from control theory, cybernetics, and cognitive systems into a single architectural language, PETRONUS provides a common substrate for reasoning about adaptation across domains. It reframes alignment not as an external constraint imposed by rewards or supervision, but as an intrinsic property maintained through structural self-consistency. In this sense, coherence becomes a first-class operational quantity, guiding not only what an agent does, but how long and under what conditions it remains itself.

While the full realization of PETRONUS will require continued development, empirical validation, and domain-specific instantiation, the architectural principles articulated here already delineate a clear path forward. They suggest a class of adaptive agents capable of sustained operation in open-ended environments without erosion of identity, meaning, or alignment. At scale, such systems point toward a form of synthetic conscience: not as an anthropomorphic attribute, but as an engineered capacity for self-preservation, contextual awareness, and coherent continuity over time.

14 Sources and Scope

Sources and Scope. This document presents a self-contained architectural synthesis of the PETRONUS framework. It defines structural principles and system-level concepts governing coherence-preserving adaptation and deliberately limits its scope to architectural exposition. Proprietary algorithms, implementation details, and patent-specific disclosures are excluded.

PETRONUS is not a speculative or purely theoretical construct. It defines a complete and internally consistent adaptive architecture intended for realization in real systems. The absence of operational detail in this document reflects a deliberate separation between architectural disclosure and implementation-level specification, rather than any conceptual incompleteness.

Certain mechanisms are withheld not because they are undefined, but because they operate at a level where premature or partial disclosure would compromise scientific rigor and responsible transfer. These aspects are reserved for collaborative work with qualified research groups under strict non-disclosure agreements.

In this sense, PETRONUS may be understood as an architecture of semantic invariants for adaptive cognition: it specifies what must remain coherent as a system learns, acts, and changes over time, independent of substrate, algorithm, or embodiment.

Patent Status. As of December 31, twenty-two provisional patent applications covering systems, methods, and structural components of the PETRONUS framework have been filed. The present document is independent of these filings and does not disclose or substitute any patent specification.

At this stage, further formalization, validation, and extension of the framework exceed the scope of individual authorship. PETRONUS is therefore open to collaborative engagement with qualified scientific and research groups for joint formal development under appropriate non-disclosure agreements.