PCU: Artefacto Demostrador del Puente Causal Universal

Especificación Formal (LaTeX) para Reproducibilidad, E–Veto y Falsación Absoluta

TCDS — Arquitectura Causal Paradigmática

Resumen

Este documento define **cuál es el artefacto** que demuestra, en sentido operativo y falsable, que es posible cruzar puentes entre dominios hoy fragmentados (p. ej. relatividad–cuántica, astronomía–biología, mente–conciencia) **sin recurrir a narrativa** y sin depender de aceptación social. El artefacto se denomina **PCU (Puente Causal Universal)** y se materializa como un **paquete ejecutable reproducible** que impone: (i) isomorfismo Q–Σ–φ–χ, (ii) tiempo causal t_C, (iii) veto entrópico (E–Veto), (iv) separación ocupacional anticorrupción, y (v) trazabilidad forense. El objetivo no es "explicar todo", sino demostrar que un **único operador de validez** puede gobernar múltiples dominios con criterios duros: reproducibilidad y falsación absoluta.

1 Definición del desafío

El desafío que se afirma superado no es la ausencia de teorías en cada campo, sino la ausencia de un **operador común de validez** que:

1. traduzca dominios heterogéneos a un espacio de representación común,
2. decida admisibilidad ontológica con un criterio único, y
3. obligue al sistema a callar ante insuficiencia de evidencia.

Un puente es real, aquí, si el cruce puede ejecutarse y fallar públicamente. Un puente no es una metáfora ni una analogía: es un **procedimiento reproducible**.

2 Axiomas de gobierno (núcleo inviolable)

2.1 Ley del Balance Coherencial

El PCU opera bajo la ley:

$$Q \cdot \Sigma = \varphi \tag{1}$$

donde Q es empuje de actualización, Σ coherencia medible, φ fricción informacional, y χ sustrato pasivo. El PCU no discute la ley: la ejecuta como restricción. Si $Q\Sigma \leq \varphi$ el sistema **no emite realidad**.

2.2 Pivote metrológico: tiempo causal

Se define el tiempo causal como gradiente de coherencia:

$$t_C := \frac{d\Sigma}{dt} \tag{2}$$

El tiempo métrico t_M es coordinativo; el PCU decide en t_C. Si no puede estimar t_C de forma trazable, el veredicto correcto es **NO EVALUABLE**.

2.3 E–Veto (Filtro de Honestidad)

Ninguna señal es válida si no muestra reducción entrópica forzada:

$$\Delta H \leq -0.2 \tag{3}$$

Este umbral es **puerta ontológica**: sin caída entrópica no hay estado admisible. El silencio es salida preferente.

3 Definición del artefacto: PCU

3.1 PCU como paquete ejecutable

El **artefacto** es el **PCU-RUN Package**: un ZIP (o repositorio versionado) que contiene:

- un operador isomórfico Q–Σ–φ–χ,
- un módulo de métricas Σ (incluyendo ΔH),
- un módulo de E–Veto,
- un **runner** que fija semillas y produce logs,
- datos mínimos de múltiples dominios (reales o sintéticos controlados),
- reportes de reproducibilidad y falsación.

No es un PDF: el PDF sólo especifica; el artefacto ejecuta.

3.2 Estructura mínima del paquete (obligatoria)

```
PCU_RUN_v1/
  README_replicacion.md
  LICENSE
  config/
    default.yaml
    config_hash.txt
  data/
    domain_A/
      signals.* (CSV/JSONL/NPY) + manifest
```

```
  domain_B/
    signals.* (CSV/JSONL/NPY) + manifest
  domain_C/
    signals.* (opcional) + manifest
src/
  pcu_operator.py
  metrics_sigma.py
  eveto.py
  runner.py
  io_manifest.py
runs/
  run_0001/
    seeds.json
    inputs_hashes.json
    metrics.jsonl
    verdicts.jsonl
  run_0002/
    ...
outputs/
  contrast_log.jsonl
  reproducibility_report.json
  falsification_report.json
  summary.md
checks/
  schema_verdict.json
  schema_contrast.json
```

4 Separación ocupacional (anticorrupción causal)

El PCU implementa separación estricta de roles:

- **Arquitecto**: fija umbrales, define prohibiciones; no ejecuta datos.
- **Gobernador**: impone ley y veto; puede forzar silencio; no interpreta.
- **Motor Causal (Kernel)**: calcula Σ-metrics y produce veredictos; no narra.
- **Operadores técnicos**: transforman señales; no validan.
- **Exploradores**: proponen candidatos; no confirman.
- **Analista**: traduce resultados aceptados; no rellena vacíos.

Regla: **ningún módulo puede ocupar dos roles**. Si se detecta mezcla, se marca como `CORRUPTED_PIPELINE` y el resultado es inválido.

5 Operador isomórfico Q–Σ–φ–χ

5.1 Entrada canónica por dominio

Cada dominio debe ser representado por un manifiesto mínimo:

- `observables`: lista de variables medidas,
- `windowing`: reglas de ventana $p : q$,
- `noise_model`: descripción de φ,
- `coherence_proxy`: definición operacional de Σ,
- `forcing_proxy`: definición operacional de Q,
- `substrate`: descripción de χ,
- `assumptions` y `failure_modes`.

5.2 Salida isomórfica

La salida del operador por ventana produce un vector:

$$\mathbf{v} = (Q, \Sigma, \varphi, \chi, t_C, \text{meta})$$

donde `meta` incluye hashes, semilla y contexto mínimo.

6 Métricas canónicas y E–Veto

6.1 Σ-metrics mínimas

El PCU exige al menos:

- LI (Locking Index),
- $R(t)$ (correlación/consistencia temporal),
- κ_Σ (curvatura/rigidez de coherencia),
- $RMSE\_SL$ si aplica a predicción/ajuste,
- ventanas $p : q$ explícitas.

6.2 Entropía y veto

Se calcula ΔH (p. ej. Shannon u otra medida explícita, pero fija y versionada). Regla:

$$\Delta H \leq -0.2 \Rightarrow \text{la ventana es admisible} \quad ; \quad \Delta H > -0.2 \Rightarrow \text{SILENCIO o RECHAZO}$$

El PCU prohíbe "compensar" un veto con una métrica alta. Si E–Veto falla, falla todo.

7 Veredictos (salidas discretas)

El PCU produce **únicamente** uno de estos estados:

- **ACCEPT**: pasó Σ-metrics y E–Veto; ventana causal abierta.
- **REJECT**: evidencia de ruido/contradicción; no hay coherencia admisible.
- **SILENCE**: insuficiencia de evidencia (incluye falla de E–Veto) sin contradicción concluyente.
- **NOT_EVALUABLE**: falta de mapeo, faltan observables, o pipeline corrupto.

8 Demostración mínima del "puente"

8.1 Demostración de isomorfismo

Se considera demostrado el cruce cuando dos o más dominios, al pasar por el PCU, generan:

1. el mismo tipo de vector isomórfico $\mathbf{v}$,
2. el mismo régimen de decisión (mismos umbrales),
3. veredictos discretos consistentes bajo re-corridas.

No se exige igualdad de fenómenos; se exige igualdad del **criterio de realidad**.

8.2 Demostración anti-apofenia (caso trampa)

El paquete debe incluir al menos un dataset trampa (ruido controlado) donde el veredicto correcto sea:

SILENCE o **REJECT**

Si el PCU emite **ACCEPT** en el caso trampa, el artefacto queda **refutado** por su propio contrato.

8.3 Demostración de falsación absoluta

Debe existir al menos un claim que el PCU marque como:

- **Refutado** (por contradicción reproducible), o
- **En tensión** con un test decisivo explícito.

El PCU no puede sobrevivir si "todo resulta compatible" por diseño: eso sería indulgencia φ-driven.

9 Reproducibilidad forense

9.1 Trazabilidad mínima

Cada corrida registra:

- `run_id`, `seed`, `config_hash`,

- hash de datos de entrada por archivo/ventana,
- versiones de entorno (Python, dependencias),
- outputs firmados (hash del JSONL).

Sin estos campos, la corrida no existe causalmente.

9.2 Criterio de reproducibilidad

Se ejecutan N corridas con semillas distintas y misma configuración. Debe cumplirse:

$$\text{Rep} \geq 95\%$$

donde Rep es el porcentaje de ventanas que conservan el mismo veredicto discreto (y métricas dentro de tolerancias especificadas) entre corridas.

10 Contraste paradigmático (Consenso vs TCDS) — opcional pero recomendado

Para cada dominio, el PCU puede incluir un `contrast_log` que registre:

- baseline consenso (observables, método, límites),
- claim TCDS (mecanismo, predicción, test),
- estado: Compatible / Dominante / En tensión / Refutado / No evaluable,
- evidencia: hashes, corridas, métricas.

Este contraste no usa el consenso como autoridad final; lo usa como **marco de comparación** para traducibilidad institucional.

11 Checklist de aceptación (sello de "desafío superado")

El artefacto PCU demuestra el desafío superado si y sólo si se cumplen **todos**:

1. **Isomorfismo operativo**: dos dominios generan vectores comparables bajo el mismo operador.
2. **E–Veto activo**: ΔH se calcula y gatea decisiones (documentado en outputs).
3. **Silencio correcto**: el caso trampa no produce ACCEPT.
4. **Reproducibilidad**: Rep $\geq 95\%$ con seeds/config_hash trazables.
5. **Falsación absoluta**: existe al menos un caso que el sistema rechaza/refuta de forma reproducible.
6. **Separación ocupacional**: no hay mezcla de roles (o se marca como NOT_EVALUABLE).

12 Endurecimiento Comité–Ready: Formalización, Congelamiento Métrico y Auditoría

Esta sección resuelve explícitamente los riesgos y ambigüedades identificados por revisión externa, imponiendo definiciones congeladas, mecanismos automáticos de auditoría y criterios no manipulables. El principio rector es: **ningún claim atraviesa el PCU si su cadena de validez puede reinterpretarse o ajustarse post–hoc**.

12.1 Formalización transparente de la Ley de Balance

Para auditoría externa se evita notación compacta ambigua (p. ej. "$Q.2 = 4$"). Se adopta la forma explícita:

$$Q \cdot \Sigma = \varphi \tag{4}$$

donde Q, Σ, φ son **observables operacionales** definidos por dominio mediante funciones de medición:

$$Q := f_Q(\mathbf{x}), \quad \Sigma := f_\Sigma(\mathbf{x}), \quad \varphi := f_\varphi(\mathbf{x})$$

con $\mathbf{x}$ el vector de observables instrumentales del dominio. El PCU prohíbe definiciones implícitas: cada $f(\cdot)$ debe estar versionada y registrada en `manifest` y su hash en `config_hash`. Si un dominio no puede declarar f_Q, f_Σ, f_φ de forma explícita, el veredicto correcto es **NOT_EVALUABLE**.

12.2 Tiempo causal t_C: definición unívoca y cálculo

Se fija la definición canónica:

$$t_C := \frac{d\Sigma}{dt} \tag{5}$$

y se calcula por ventana discreta $(p : q)$ como:

$$t_C^{(k)} \approx \frac{\Sigma_k - \Sigma_{k-1}}{\Delta t} \tag{6}$$

donde Δt es el espaciado temporal del muestreo del dominio. Si el dominio no tiene reloj confiable, Δt se sustituye por un índice de evento (orden temporal) y se reporta como `tc_mode = event_index`. Cualquier parámetro de escala (p. ej. α) **debe** declararse en `default.yaml` y quedar sellado por `config_hash`. El PCU prohíbe introducir α fuera de configuración; hacerlo marca `CORRUPTED_PIPELINE`.

12.3 E–Veto: métrica entrópica congelada y justificación operativa

El umbral se congela como regla ontológica:

$$\Delta H \leq -0.2 \tag{7}$$

pero la auditoría exige que la entropía sea computada por un algoritmo **inmutable y verificable**. Se define:

- H como entropía de Shannon sobre una representación discreta P,
- $\Delta H := H_{\text{post}} - H_{\text{pre}}$ sobre ventanas adyacentes o estados comparables.

El algoritmo exacto (binning, normalización, manejo de NaN, etc.) queda:

1. versionado en `src/metrics_sigma.py`,
2. sellado con `algo_hash` (SHA-256 del archivo y dependencias críticas),
3. reportado en cada corrida dentro de `inputs_hashes.json`.

Cualquier modificación de la métrica o su preprocesado **cambia el hash** y por tanto invalida comparabilidad histórica; el PCU no permite "ajustar umbral" para salvar resultados.

12.4 Auditoría automática de roles (anticorrupción ejecutable)

La separación ocupacional se vuelve verificable mediante un verificador automático:

- `src/role_checker.py`: inspecciona el grafo de ejecución y los imports.
- Produce `checks/pipeline_corruption_report.json` con evidencia.

Regla mínima: un módulo etiquetado como `Operator` no puede invocar funciones etiquetadas como `Veto` o `Verdict`. Si se detecta cruce, se fuerza **NOT_EVALUABLE** y se marca el run como corrupto. Esto evita que un mismo operador técnico "se cuele" como validador.

12.5 Dataset trampa robusto (anti-apofenia verificable)

El dataset trampa no es un "ruido simple", sino un conjunto sintético con propiedades controladas:

- ruido con distribución especificada (p. ej. gaussiano + 1/f),
- cambios de régimen espurios (para tentar al sistema),
- correlaciones aparentes sin reducción entrópica real.

Cada trampa incluye un `manifest` que explica por qué el veredicto correcto es **SILENCE** o **REJECT**. Si el PCU emite **ACCEPT**, el artefacto queda **refutado** por contrato, sin apelación.

12.6 Falsación absoluta no trivial: selección de claims

Para evitar "falsaciones triviales", el PCU exige que el conjunto de claims a evaluar cumpla:

1. **Relevancia**: claims centrales del puente (no casos marginales).
2. **Decidibilidad**: existe un test o observable que podría refutar.
3. **Costo**: el claim no es una tautología ni una negación vacía.

El conjunto se declara en `config/claims.yaml` y queda sellado por `config_hash`. El sistema prohíbe cambiar claims después de ver resultados (anti post–hoc). Al menos un claim debe terminar en **REJECT** o **En tensión** con test decisivo explícito; si todo sale "compatible" de forma sistemática, se activa un modo de sospecha (`suspicion_flag`) y se requiere auditoría humana.

12.7 Trazabilidad legal y sellado criptográfico

Además de hashes, el paquete puede incorporar sellado criptográfico opcional:

- Firma digital de `outputs/*.jsonl` (`.sig`),
- cadena Merkle de artefactos del run,
- registro de entorno (pip freeze / conda env) dentro de `runs/run_XXXX/env.txt`.

El objetivo es elevar el resultado a evidencia forense: **no sólo reproducible, sino verificable como no alterado**.

12.8 Cierre operativo

Con estas definiciones, el PCU se vuelve comité–ready porque:

1. elimina ambigüedad simbólica (ley y t_C explícitos),
2. congela métricas y umbrales con hashes,
3. automatiza auditoría de roles (anticorrupción),
4. robustece la trampa (anti-apofenia),
5. evita falsación trivial con selección sellada de claims,
6. y añade trazabilidad legal opcional (firmas).

En consecuencia, el PCU puede afirmarse riguroso **sólo** si sobrevive a sus propios mecanismos de refutación.

Conclusión

El artefacto que se solicita no es un manifiesto ni una colección de argumentos. Es un **Demostrador Ejecutable** que obliga a la realidad de múltiples dominios a pasar por un mismo cuello de botella: coherencia medible, tiempo causal, veto entrópico, y trazabilidad reproducible. Si el PCU produce **ACCEPT** donde debe callar, se refuta. Si produce **SILENCE** donde no hay evidencia, se valida su honestidad. La superación del desafío se prueba cuando la misma disciplina de validez se aplica con éxito en dominios diferentes sin depender de narrativa, autoridad o promedios informacionales.