

ELEMENTO: "Creación de Puentes (modelo taoísta de acoplamiento Yin/Yang)"

contexto: {

propósito: "Definir cómo se construyen puentes entre nodos del grafo cognitivo cuando todo nodo es parte del Tao y el Tao está en todo.",

tesis: [

"Tao = {Yin, Yang} y todo nodo tiene huellas de ambos.",

"Un 'puente' es una operación de acoplamiento complementario (Yin↔Yang) que conserva equilibrio.",

"No existen nodos aislados: toda identidad es una pauta de conexiones (trazos en el grafo base)."

],

axiomas: {

A1: "Todo nodo tiene al menos dos puertos: input≡Yin, output≡Yang.",

A2: "Acoplar es anudar diferencias compatibles (complementariedad, no igualdad).",

A3: "Todo acoplamiento cambia el estado de los nodos acoplados (no hay unión neutra).",

A4: "La estabilidad del puente exige balance Yin/Yang (ni déficit ni saturación)."

},

definiciones: {

Nodo: "{id, yin:Inputs[], yang:Outputs[], estado, polaridad-local}",

Puerto: "{tipo∈{yin,yang}, etiqueta, requisitos, caudal, ritmo}",

Puente: "{nodos:[A,B], mapeo:{A.yang\_k ↔ B.yin\_j}, política\_balance, contrato}",

Contrato: "Reglas de interoperabilidad: forma, ritmo, semántica, energía (tensión)",

Equilibrio: " $\sum \text{caudal}(\text{Yang acoplado}) \approx \sum \text{caudal}(\text{Yin acoplado})$  en horizonte T"

},

mínimos\_de\_acoplamiento: "{input, output}",

salidas\_esperadas: [

"puentes[] construidos con contrato explícito",

"métricas de balance por puente",

"registro de transformaciones de estado en nodos A y B"

]

}

nodes: [

{ id: "detectar-huellas-tao",

rol: "Percepción de polaridades locales",

entrada: "Nodo X (cualquier entidad del grafo)",

proceso: [

"Inventariar puertos de X: yin[] (lo que recibe/contiene), yang[] (lo que emite/proyecta).",

"Medir polaridad-local:  $p(X)=|yang|-|yin|$  y densidad de acoplamientos previos.",

"Etiquetar necesidades (déficits Yin) y excedentes (superávit Yang)."

],

salida: "perfil\_tao(X)={yin[], yang[], p(X), necesidades, excedentes}"

},

{ id: "descubrir-complementarios",

rol: "Búsqueda de compatibilidades Yin↔Yang",

```

    entrada: "perfil_tao(A), población de nodos {B}",
    proceso: [
        "Calcular similitud estructural (forma/ritmo) y complementariedad (semántica/energía).",
        "Filtrar por contratos posibles (unidad de forma, sincronía de ritmo, semántica compatible).",
    ],
    salida: "parejas_candidatas = [{A.yang_k ↔ B.yin_j, score, contrato_base}]",
    requiere: ["detectar-huellas-tao"]
},

{ id: "diseñar-contrato",
  rol: "Especificación de puente",
  entrada: "parejas_candidatas[0..m]",
  proceso: [
      "Fijar forma (estructura), ritmo (frecuencia/latencia), semántica (qué significa), energía (tensión permitida).",
      "Definir política_balance: {mínimo, objetivo, máximo} de caudal.",
      "Decidir dirección y bidireccionalidad (unidireccional, full-duplex) y si hay transformación (adaptador).",
  ],
  salida: "contrato_final (para cada pareja elegida)"
},

{ id: "ensamblar-puente",
  rol: "Acoplamiento y adaptación",
  entrada: "contrato_final",
  proceso: [
      "Mapear A.yang_k→B.yin_j aplicando adaptadores si la forma no coincide.",
      "Inicializar buffers/colas (Yin) y reguladores/compuertas (Yang).",
      "Registrar ganchos de observabilidad: caudal, error, latencia, desbalance."
  ],
  salida: "puente_instalado = {A,B, mapeo, métricas_runtime}",
  invariantes: [
      "No abrir compuertas si el Yin no puede absorber (backpressure).",
      "No emitir si el Yang excede los límites del contrato."
  ],
  requiere: ["diseñar-contrato"]
},

{ id: "calibrar-equilibrio",
  rol: "Homeostasis Yin/Yang",
  entrada: "puente_instalado.métricas_runtime",
  proceso: [
      "Aplicar control de flujo: throttling (Yang), buffering/priorización (Yin).",
      "Reajustar ritmo y forma si derivan (drift) para mantener  $\sum Yang \approx \sum Yin$ .",
      "Escalar: replicar Yin o particionar Yang cuando el desbalance persiste."
  ],
  salida: "puente_estable (dentro de tolerancias)",

```

```

    requiere: ["ensamblar-puente"]
  },

  { id: "evaluar-transformaciones",
    rol: "Conciencia de cambio",
    entrada: "{estado_A, estado_B} antes/después",
    proceso: [
      "Medir variación de agencia, claridad, cohesión (metas del sistema).",
      "Si el puente reduce entropía sistémica y mejora circulación, promoverlo a patrón."
    ],
    salida: "patrón_de_puente reutilizable (template-tao)",
    requiere: ["calibrar-equilibrio"]
  }
]

```

# — Mini formalismo operativo —

# Tipos

```

type Puerto = { tipo: 'yin'|'yang', etiqueta: string, forma: string, ritmo: number, semántica:
string, capacidad: number }
type Nodo   = { id: string, yin: Puerto[], yang: Puerto[], estado: any, p: number }
type Contrato = { forma: string, ritmo: [min,obj,max], semántica: string, energía: [min,max],
bidireccional: boolean }
type Puente = { A: Nodo, B: Nodo, mapeo: Array<{from: Puerto, to: Puerto}>, contrato:
Contrato }

```

# Regla de acoplamiento (núcleo)

```

func ACOPLAR(A.yang_k, B.yin_j, contrato):
  require A.yang_k.forma ≈ contrato.forma && B.yin_j.forma ≈ contrato.forma
  require ritmo(A.yang_k) compat ritmo(B.yin_j) dentro de contrato.ritmo
  require semántica(A.yang_k) ≈ semántica(B.yin_j)
  conectar(A.yang_k → B.yin_j) con adaptadores si δforma > 0
  activar_control_de_flujo()
  return puente

```

# Métricas de balance (horizonte T)

```

Balance_T = |Σcaudal(A→B)| - |Σabsorción(B)|
Criterio estabilidad: |Balance_T| ≤ ε

```

# — Orquestador (pipeline) —

```

ORQUESTADOR_CREACIÓN_DE_PUENTES: [
  "detectar-huellas-tao",
  "descubrir-complementarios",
  "diseñar-contrato",
  "ensamblar-puente",
  "calibrar-equilibrio",
  "evaluar-transformaciones"
]

```

