```
{
  "id": "cnode:lenguaje-corp/cognitive-module.from-code@1.0.0",
  "C": {
    "orchestrator": "Toma un proyecto de código y lo convierte en un Módulo Cognitivo integrable: inventaría el repo, extrae capacidades reales, firma la estructura (partes/relaciones/orden), modela la orquestación, compone interfaces públicas (PROVIDES) y dependencias (REQUIRES), fija invariantes y políticas, define métricas y tests, y emite el módulo con su CNR, perfiles de integración y trazabilidad.",
    "nodes": [
      { "nombre": "inventory.scan", "rol": "Leer árbol del proyecto y artefactos clave (packages, entrypoints, servicios, pipelines)", "entrada": "ruta del repo / snapshot", "salida": "CodeInventory" },
      { "nombre": "capability.extract", "rol": "Identificar capacidades reales (APIs, workers, jobs, pipelines) y flujos de datos", "entrada": "CodeInventory", "salida": "CapabilitiesMap" },
      { "nombre": "structure.signature", "rol": "Derivar firma estructural del sistema (PARTS, RELS, ORDER, LAYERS) para preservar composición/adyacencia/orden", "entrada": "CapabilitiesMap", "salida": "StructureSignature" },
      { "nombre": "orchestration.model", "rol": "Modelar el proceso núcleo como orquestador (fases, entradas/salidas, estados)", "entrada": "StructureSignature", "salida": "OrchestratorSpec" },
      { "nombre": "interfaces.compose", "rol": "Definir export_surface (PROVIDES) y dependencias (REQUIRES) con tipos canónicos", "entrada": "OrchestratorSpec", "salida": "InterfaceSpec {provides[], requires[], types}" },
      { "nombre": "constraints.set", "rol": "Fijar invariantes y políticas (composition/adjacency/ordering, modality-minimum, provenance, umbrales)", "entrada": "InterfaceSpec", "salida": "ConstraintsPolicy" },
      { "nombre": "metrics.define", "rol": "Definir métricas/quality signals (dominio + Ω/C/Λ/Φ si aplica) y umbrales de aceptación", "entrada": "ConstraintsPolicy", "salida": "MetricsSpec" },
      { "nombre": "cnr.emit", "rol": "Construir la representación CNR (C = {orchestrator, nodes[]}) del módulo", "entrada": "OrchestratorSpec + InterfaceSpec + ConstraintsPolicy + MetricsSpec", "salida": "CNR" },
      { "nombre": "integration.profile", "rol": "Generar perfil Always-Integratable (adapters, negotiation, merge_policy, id/versionado)", "entrada": "CNR + InterfaceSpec", "salida": "IntegrationProfile" },
      { "nombre": "tests.plan", "rol": "Listar casos de prueba (unit/contract/e2e) y casos de borde para validar invariantes y métricas", "entrada": "CNR + MetricsSpec", "salida": "TestPlan" },
      { "nombre": "provenance.builder", "rol": "Armar traza de procedencia (decisiones, mapeos, archivos origen → nodos)", "entrada": "CNR + IntegrationProfile", "salida": "ProvenanceTrace" },
      { "nombre": "module.emitter", "rol": "Emitir el Módulo Cognitivo final listo para integrar (CNR + perfiles + schemas + stubs)", "entrada": "CNR + IntegrationProfile + TestPlan + ProvenanceTrace", "salida": "CognitiveModulePackage" }
    ]
  }
}
```