Catch-All/Connector Layers

The clean-up crew of your platform

- Not every layer fits a clean category. Catch-all layers are hybrids that do complementary work for multiple other layers. They handle tasks that don't belong to any single specialized layer but are essential for the system to function cohesively

- These layers often emerge when you discover gaps like two layers need to work together but speak different "languages," or several layers all need the same preprocessing that none of them should be responsible for individually

- In Emstrata, the Chron-Con does more than just error correction. It also tracks secrets and memories from the narrative, explicitly tagging them for Groundskeeper to integrate into system memory. You don't want Narration burdened with the unrelated task of extracting and categorizing secrets while it's trying to write high-quality prose. And Groundskeeper needs these pieces explicitly labeled as "secrets" or "memories" to properly integrate them into the simulation history. The Chron-Con bridges this gap.

Cyclical Vs Circumstantial Systems

And everything in-between

- Cyclical systems run the same prompts every time. Emstrata follows this pattern: every turn runs Discovery, then Narration, then Chron-Con, then Groundskeeper, in that exact order. The flow is predictable and consistent regardless of what happens in the simulation. You always know what's executing next, which makes debugging straightforward and cost estimation more reliable

- Circumstantial systems determine the pathway based on outcomes or AI direction. The route through your architecture changes depending on what happened in previous steps. Maybe an error detection layer decides whether correction is needed. Maybe a routing layer examines user intent and sends the request down completely different processing paths. The system adapts its own execution flow based on runtime conditions

- Hybrid systems are mostly cyclical at their base, but circumstantial at times when specific conditions warrant different handling. You might always run your core cycle, but branch to specialized subsystems when certain triggers fire. Many real-world systems end up here. It’s a reliable backbone with conditional branches for edge cases. Emstrata has a number of circumstantial offshoots as well