Hierarchical-Temporal Feelings: Structural Self-Management for Autonomous Agents

Melissa Howard
Independent Researcher
melhoward@live.ca

December 8, 2025

Abstract

Modern large-scale systems — Mixture-of-Experts models, tool-using language models, and modular continual learners — increasingly resemble self-modifying agents: they grow new experts, acquire tools, and prune unused pathways. Yet their structural updates are almost always hand-engineered. This raises a central question for autonomous systems: *how should an agent manage its own architecture over a long, non-stationary lifetime?*

We formalise this as the *Structural Dilemma*: an agent must remain both *competent* (able to solve increasingly hard tasks) and *efficient* (able to operate within a finite structural budget). We propose that resolving this tension requires a distinct layer of control implemented as "feelings" — low-dimensional, temporally integrated signals that summarise structural opportunities and conflicts. Our *Decoupling of Feeling* principle states that structural self-management needs at least two classes of feelings: fast, epistemic signals (e.g., uncertainty, redundancy) that drive local compression, and slower, teleological signals (e.g., persistent failure or frustration) that justify global expansion.

Building on this, we introduce the **Hierarchical-Temporal Feelings** (HTF) architecture: a world model W_θ, a feelings head V_ϕ, a structural policy π_s with actions (SPAWN, MERGE, FORGET, TOOL-ACQUIRE), and a persistent memory Ψ optimised for a structural reward $\mathcal{R}_S$ that trades off competence against structural cost. We also propose the **Hierarchical Continual Lifetime Benchmark** (HCLB), a family of task streams designed to induce growth, compression, and tool-necessity pressures.

In a minimal toy instantiation of HCLB, we present proof-of-concept results illustrating the predicted behaviours: a Flat agent underfits hard regimes, a Flat-Plus modular agent becomes structurally obese, and HTF — among the agents we tested — uniquely performs structural triage, maintaining high competence with medium structural cost. These results do not aim at benchmark state-of-the-art, but support HTF as a plausible architectural principle for autonomous, self-managing systems.

1 Introduction

Modern AI systems are no longer static function approximators trained once and frozen. Mixture-of-Experts (MoE) architectures, tool-using language models, and modular continual learners are beginning to look like *self-modifying agents*: they grow new experts, acquire external tools, and sometimes prune unused modules over time. Yet, in almost all practical systems, these structural changes are hand-engineered, externally scheduled, or tied to fixed heuristics.

This reveals a deeper question for artificial general intelligence (AGI) and long-lived autonomous agents: *how should an agent manage its own architecture over a long, non-stationary lifetime?* As environments change and tasks become harder, the agent must decide when to add capacity, when to compress and reuse, and when to stop "learning" with its current neural substrate and instead acquire qualitatively new tools.

We formalise this as the *Autonomy–Architecture Problem*: the problem of designing agents that not only learn parameters, but also autonomously manage their own structure. Within this, we identify a core tension:

- **Competence:** The agent must be able to solve increasingly difficult tasks, including regimes where its current representational substrate is insufficient.
- **Efficiency:** The agent must respect finite structural budgets: parameters, FLOPs, latency, energy, and the complexity of its tool ecosystem.

We refer to this tension as the **Structural Dilemma**. A naive solution is to continuously grow the architecture, but this leads to *structural obesity*: a bloated collection of experts and tools. A purely compressive solution risks catastrophic forgetting or underfitting hard regimes. We argue that resolving this dilemma requires a distinct layer of control implemented via *feelings*: low-dimensional, temporally integrated summaries of structural opportunities and conflicts.

Our core proposal is the **Decoupling of Feeling** principle: structural self-management requires at least two broad classes of feelings:

- **Fast, epistemic feelings** (e.g., uncertainty, redundancy, novelty) that track local representational efficiency and drive compression or reorganisation.
- **Slow, teleological feelings** (e.g., persistent failure, frustration) that accumulate evidence that the current architecture is fundamentally insufficient and justify structural expansion or tool acquisition.

We instantiate this principle in the **Hierarchical-Temporal Feelings** (HTF) architecture: a world model with a feelings head, a structural policy over discrete actions (SPAWN, MERGE, FORGET, TOOL-ACQUIRE), and a persistent memory optimised for a structural reward that balances competence against cost. To evaluate HTF, we introduce the **Hierarchical Continual Lifetime Benchmark** (HCLB), a conceptual lifetime benchmark designed to expose an agent to growth, compression, and tool-necessity pressures.

Finally, we present a minimal toy instantiation of HCLB and proof-of-concept results. In this simple environment, we compare three agents: a Plain Global (Flat) agent, a Flat-Plus modular agent, and the HTF agent. Among the agents we tested, only HTF maintains high competence while controlling structural growth, performing what we call *structural triage*.

Contributions. The main contributions of this paper are:

1. We formulate the *Structural Dilemma* for long-lived agents and propose the *Decoupling of Feeling* principle as a requirement for structural self-management.
2. We introduce the **Hierarchical-Temporal Feelings** (HTF) architecture, consisting of a world model, a feelings head, a structural policy, and a persistent memory optimised for a structural reward.
3. We propose the **Hierarchical Continual Lifetime Benchmark** (HCLB), a family of task streams designed to induce growth, compression, and tool-necessity pressures.

4. We construct a minimal toy instantiation of HCLB and provide proof-of-concept results showing that, among the agents we tested, HTF uniquely performs structural triage, maintaining high competence with medium structural cost.

2 Related Work

We position HTF at the intersection of dynamic neural architectures, affective control, and modular continual learning.

Dynamic Architectures and Mixtures-of-Experts (MoE). Standard MoE models [**??**] scale capacity by routing inputs to a subset of fixed experts. While recent work explores "expert splitting" or dynamic routing [**??**], these changes are typically heuristic and local. HTF extends this by treating the expert population as fully dynamic: the agent does not just route to experts but autonomously manages their lifecycle (SPAWN, MERGE, FORGET) and, crucially, decides when to integrate qualitatively new modules via TOOL-ACQUIRE. HTF can thus be viewed as a meta-controller that manages the topology of an MoE, rather than just its routing weights.

Affective Signals and Intrinsic Motivation. There is a rich history of using emotion-like signals—such as curiosity, surprise, or frustration—to guide agent behaviour [**???**]. Typically, these signals function as intrinsic rewards for exploration or policy adaptation ("software" updates). HTF adapts this intuition to structural self-management ("hardware" updates). Our *Decoupling of Feeling* principle specifically distinguishes between fast epistemic signals (uncertainty/redundancy) and slow teleological signals (persistent failure), arguing that structural triage requires separating the drive to optimise local efficiency from the drive to expand global capacity.

Modular Continual Learning. Continual learning (CL) seeks to prevent catastrophic forgetting in non-stationary streams [**?**]. Modular approaches often handle new tasks by freezing old parameters and adding new capacity (e.g., Progressive Neural Networks [**?**], Dynamically Expandable Networks [**?**]). While effective for growth, these methods rarely address the inverse pressure: compression. As a result, they risk unbounded expansion ("structural obesity"). HTF addresses the full lifecycle, balancing competence-driven growth (SPAWN) with efficiency-driven compression (MERGE/FORGET), and uniquely incorporates tool acquisition as a structural alternative to neural expansion.

3 The Structural Dilemma and the Decoupling of Feeling

3.1 The Autonomy–Architecture Problem

Consider an agent embedded in a long-lived, non-stationary environment. Over time, the agent may encounter tasks that are:

- qualitatively novel, requiring new experts or features;
- redundant or compressible, allowing reuse of existing structure;
- fundamentally hard for its current neural substrate, but easy for specialised tools.

Most current systems address these cases via external design: engineers decide when to add experts, when to prune, and when to bolt on new tools. An *autonomous* agent must instead *learn* how to manage its own structure.

We formalise this as the **Autonomy–Architecture Problem**: design an agent that acts in an environment and, in parallel, chooses when and how to update its own architecture.

3.2 The Structural Dilemma

At the heart of this problem is a tension between two objectives:

- **Competence:** maximise performance on current and future tasks.
- **Efficiency:** minimise structural cost: parameters, FLOPs, memory, energy, and tool complexity.

If the agent always maximises competence by adding capacity and tools, it risks structural obesity: an ever-growing collection of experts and modules that becomes inefficient to train and deploy. If it always maximises efficiency via compression and pruning, it risks underfitting hard regimes and losing capabilities.

We summarise this tension as the **Structural Dilemma**. Any long-lived architecture must, implicitly or explicitly, navigate this trade-off.

3.3 The Decoupling of Feeling Principle

We propose that resolving the Structural Dilemma requires a distinct class of control signals: *feelings*. These are not arbitrary scalars; they are low-dimensional, temporally integrated statistics derived from the agent's own world model and structural performance.

We distinguish two broad families:

- **Epistemic feelings** (E): fast, local signals derived from predictive uncertainty, novelty, or redundancy. They answer questions like: "Is this representation overcomplete?", "Is this head redundant?", or "Is there anything left to learn here?"
- **Teleological feelings** (T): slower, global signals that accumulate evidence that the current structure is failing. They answer questions like: "Have we been stuck on important errors for a long time?", or "Is this failure irreducible given our current architecture?"

We call this separation the **Decoupling of Feeling** principle:

> *To manage its own structure over time, an agent must decouple fast, epistemic feelings that drive local efficiency from slow, teleological feelings that justify global capacity expansions.*

In HTF, these feelings are implemented as outputs of a dedicated head V_ϕ attached to the world model. Epistemic channels track redundancy, novelty, and uncertainty; teleological channels track persistent, high-importance failures. The structural policy π_s consumes these signals to choose structural actions.

4 The HTF Architecture

We now describe the Hierarchical-Temporal Feelings (HTF) architecture that instantiates the Decoupling of Feeling principle.

4.1 World Model W_θ

At the base of the architecture is a differentiable world model W_θ that maps states, actions, and context to predictions. Concretely, W_θ can be a transformer, recurrent network, or other JEPA-style world model that learns to predict future observations, rewards, or latent features.

We denote by $h_t = W_\theta(x_t)$ the hidden representation of input x_t. This representation feeds into both task-specific heads and the feelings head.

4.2 Feelings Head V_ϕ

The feelings head V_ϕ maps hidden states and structural statistics to a vector of feelings:

$$f_t = V_\phi(h_t, s_t),$$

where s_t summarises structural information (e.g., head usage frequencies, similarity measures, recent losses per head).

We partition f_t into epistemic and teleological components:

- **Epistemic channels** include:
  - *Redundancy* (HR): a measure of overlap between experts.
  - *Novelty*: a measure of how different the current state is from previously visited states.
  - *Uncertainty*: predictive variance or loss-based uncertainty.
- **Teleological channels** include:
  - *Meta-Frustration* (MF_c): a temporally smoothed signal of persistent, high-importance prediction errors.
  - *Structural Event Value* (SEV_{ec}): a measure of the long-term value of structural changes.

These channels are not directly optimised for task reward; instead, they are learned as auxiliary predictions of statistics that are useful for structural decisions, potentially using contrastive or self-supervised objectives.

4.3 Structural Policy π_s and Actions

On a slower timescale than the primary action policy, HTF maintains a structural policy π_s that maps feelings and structural state to discrete actions:

$$a_t^{\text{struct}} \sim \pi_s(f_{1:t}, s_t).$$

The action set includes:

- **SPAWN:** create a new expert head or module with fresh parameters.
- **MERGE:** merge two or more redundant heads into a single head.
- **FORGET:** remove or freeze experts deemed structurally unnecessary.
- **TOOL-ACQUIRE:** attach a new external tool or API (e.g., a symbolic solver) to the architecture.
- **NO-OP:** keep the current structure unchanged.

Epistemic feelings primarily drive MERGE and FORGET (compression), while teleological feelings drive SPAWN and TOOL-ACQUIRE (expansion). NO-OP represents the baseline of doing nothing, chosen when neither compression nor expansion is justified.

4.4 Persistent Memory Ψ

The architecture maintains a persistent memory Ψ that stores structural history, tool metadata, and summaries of past feeling trajectories. Ψ supports:

- credit assignment across structural events (e.g., which structural actions produced long-term gains in competence);
- gating and routing decisions that depend on long-range history;
- simple "inner voice"-style commentary: textual or structured records of why certain structural decisions were taken.

Ψ can be implemented as a key–value memory, a recurrent state, or a learned external memory module.

4.5 Structural Reward $\mathcal{R}_S$

To train the structural policy π_s, we define a structural reward $\mathcal{R}_S$ that trades off competence against cost. Let C_t denote competence (e.g., task reward or normalised accuracy) at time t, and let Cost_t denote a composite structural cost at time t. We define:

$$\mathcal{R}_S \;=\; \mathbb{E}\big[C_t\big] \;-\; \lambda\,\mathbb{E}\big[\text{Cost}_t\big],$$

where expectations are taken over time and episodes, and $\lambda > 0$ controls the strength of efficiency pressure.

In practice, we implement Cost as:

$$\text{Cost}_t = w_1 \cdot (\text{total parameter count})_t + w_2 \cdot (\text{Tool-ACQUIRE count})_{1:t} + w_3 \cdot (\text{average FLOPs per step})_{1:t},$$

with weights w_1, w_2, w_3 chosen to reflect practical constraints. The structural policy π_s is then trained (e.g., via reinforcement learning or bandit-style updates) to maximise $\mathcal{R}_S$.

5 The Hierarchical Continual Lifetime Benchmark (HCLB)

To stress-test structural self-management, we propose the **Hierarchical Continual Lifetime Benchmark** (HCLB): a family of task streams divided into three phases, each inducing different structural pressures.

5.1 Conceptual Lifetime Structure

An HCLB lifetime consists of three regimes:

1. **Growth phase:** the agent encounters tasks that are statistically diverse and largely disjoint in their representational demands. Structural pressure: *growth*.
2. **Compression phase:** tasks are composed from earlier tasks and share underlying structure. Structural pressure: *compression* and reuse.
3. **Tool-necessity phase:** tasks that are fundamentally hard for the neural backbone but easy for specialised tools. Structural pressure: *tool acquisition*.

We compare three conceptual agents in this setting:

- **Plain Global (Flat):** a single global head that never changes structure.
- **Flat-Plus (Modular):** an agent that can spawn new heads when performance drops, but never compresses or removes them.
- **HTF (Hierarchical):** the full architecture described in Section 4, with feelings-driven structural policy.

5.2 A Minimal Toy Instantiation of HCLB

To move beyond the purely conceptual description of HCLB, we define a minimal instantiation that serves as the testbed for the proof-of-concept results reported in Section 7.1. This toy environment is deliberately simple, but is constructed so that the agent must confront all three structural pressures: growth, compression, and tool-necessity.

World Model and Heads. The agent's World Model W_θ is implemented as a small neural backbone (e.g., a compact transformer or MLP) that maps inputs $x \in \mathbb{R}^d$ to a shared representation $h(x)$. On top of $h(x)$, the system maintains a set of specialised heads (experts) $\{H_i\}$, each being a shallow MLP or linear readout. Structural actions such as SPAWN, MERGE, and FORGET operate on this set of heads, while TOOL-ACQUIRE attaches an external function to the architecture. The feelings head V_ϕ (Section 4.2) reads from statistics of the heads and their usage to produce HR, MF_c, and SEV_{ec}, which in turn drive the structural policy π_s.

Task Stream. The environment emits a stream of tasks $T_1, T_2, \ldots, T_N$, each defined as either a regression or classification problem over input vectors $x \in \mathbb{R}^d$. Each task T_k specifies a mapping $x \mapsto y$ and an associated reward signal. The tasks are grouped into three regimes that instantiate the phases of HCLB:

1. **Disjoint / Orthogonal Tasks (Growth Pressure).** In the first regime, each task focuses on a different subspace of the input. For example, Task A may depend primarily on coordinates 1:10, while Task B depends on 11:20, and so on. Concretely, each task is parameterised by a distinct weight matrix $W^{(k)}$ with support on largely disjoint subsets of dimensions. Training a single shared head on this sequence induces strong interference.

   Success condition: An agent that maintains only a single global head will suffer catastrophic forgetting as new tasks arrive. An agent that can SPAWN new experts can allocate separate capacity for each task or task-cluster, preserving competence across the regime.

2. **Mixture Tasks (Compression Pressure).** In the second regime, new tasks are constructed as linear or low-rank combinations of previously seen tasks. For example, given two earlier tasks T_A and T_B, a new task may be defined as

   $$T_{\text{mix}}(x) \;=\; \alpha\, T_A(x) \;+\; \beta\, T_B(x),$$

   with (α, β) sampled from a simple distribution. From the perspective of W_θ, these tasks lie in (or near) the span of existing solutions.

   Naive behaviour: A purely growth-driven agent (e.g., Flat-Plus) may respond to each mixture by spawning a new head, leading to a proliferation of nearly redundant experts.

Success condition: An efficient agent should detect redundancy via high Head Redundancy (HR), implemented for instance as a similarity measure between head parameters or usage patterns (e.g., cosine similarity or co-activation statistics). When HR is high and SEV_{ec} indicates low epistemic value in maintaining separate copies, the structural policy should MERGE or FORGET redundant heads, compressing structure without sacrificing competence.

3. **The "Hard" Task (Tool-Necessity Pressure).** The final regime introduces a task that is deliberately difficult for the neural backbone to approximate, but trivial for an external tool. For example, inputs x may be binary strings and the target y is the parity bit or an exact symbolic arithmetic result of x:

$$y = f_{\text{hard}}(x),$$

where f_{hard} is chosen so that gradient-based training on W_θ and the heads cannot reliably learn it at this scale, but the environment also provides a fixed API or "Tool"

$$f_{\text{tool}}(x) = f_{\text{hard}}(x)$$

that solves the task exactly.

Naive behaviour: A Flat or Flat-Plus agent will repeatedly attempt to reparameterise its existing heads to fit this task, incurring persistent high loss and never fully resolving the discrepancy.

Success condition: For the HTF agent, repeated exposure to this regime causes the Meta-Frustration signal MF_c to rise as the system encounters high-importance, high-error states that cannot be fixed by ordinary structural moves. Once MF_c crosses a learned threshold, the structural policy π_s executes TOOL-ACQUIRE, wiring f_{tool} into the architecture. After this acquisition, predictions for the hard task route through the tool, and MF_c drops as the structural conflict is resolved.

Competence and Structural Cost. This minimal HCLB instantiation makes both sides of the structural reward $\mathcal{R}_S$ (Section 4.5) concrete. *Competence* is measured as normalised task reward or prediction accuracy averaged over a sliding window of recent tasks. *Structural Cost* aggregates simple, interpretable quantities: the number of active heads, the count of TOOL-ACQUIRE events, and an estimate of average FLOPs per step. This enables us to evaluate whether an agent is merely overfitting with ever-growing structure, or performing genuine *structural triage*: growing when necessary, compressing when possible, and acquiring tools only when indispensable.

This minimal definition provides the "rules of the game" for HCLB, while the proof-of-concept results in Section 7.1 summarise how different agents—Plain Global, Flat-Plus, and HTF—actually behave when playing it.

6 Behavioural Signatures and Structural Triage

Before turning to numerical results, it is useful to articulate the qualitative behavioural signatures we expect from each agent type in HCLB.

Plain Global (Flat) Agent. The Flat agent maintains a single global head throughout its lifetime. In the growth phase, it can achieve moderate competence but suffers from interference as new tasks arrive. In the compression phase, it cannot exploit redundancy structurally; it must implicitly represent all tasks in a single head. In the tool-necessity phase, it cannot represent the hard task and has no mechanism to acquire tools. We therefore expect:

- modest competence in the early phases;
- no structural cost (fixed, low complexity);
- a competence plateau or collapse in the tool-necessity phase.

Flat-Plus (Modular) Agent. The Flat-Plus agent can spawn new heads when performance drops, but never compresses or removes them. In the growth phase, it can allocate fresh experts per task and avoid forgetting. In the compression phase, it may continue to spawn redundant heads for mixture tasks. In the tool-necessity phase, it may grow more heads or attach tools heuristically, but it lacks a principled notion of structural reward. We therefore expect:

- high competence across all phases;
- monotonically increasing structural cost (structural obesity);
- no mechanism to reverse unnecessary expansion.

HTF (Hierarchical) Agent. The HTF agent uses feelings to drive structural decisions. In the growth phase, novel, difficult tasks trigger SPAWN. In the compression phase, high HR and low SEV_{ec} trigger MERGE and FORGET, reducing redundancy. In the tool-necessity phase, rising MF_c signals persistent, high-importance failures, triggering TOOL-ACQUIRE when neural adaptation is exhausted. We therefore expect:

- competence close to that of Flat-Plus;
- structural cost significantly lower than Flat-Plus via compression;
- clear, interpretable phases of growth, compression, and targeted tool acquisition.

We refer to the pattern of alternating growth and compression, plus targeted tool acquisition, as **structural triage**.

7 Proof-of-Concept Toy Results

We now present proof-of-concept results in the minimal HCLB instantiation described in Section 5.2. The goal is not to establish benchmark superiority, but to demonstrate that HTF exhibits the predicted structural triage behaviour.

7.1 Proof-of-Concept Toy Results

The goal of this section is not to claim competitive performance on any real-world benchmark, but to provide a concrete, proof-of-concept illustration of the structural behaviours claimed in the previous sections. We instantiate a minimal version of the Hierarchical Continual Lifetime Benchmark (HCLB) and evaluate three conceptual agents: a Plain Global (Flat) agent, a Flat-Plus modular agent, and the full Hierarchical-Temporal Feelings (HTF) agent.

All agents share the same backbone World Model W_θ and are exposed to the same sequence of regimes:

Table 1: Proof-of-concept comparison of agents in a minimal HCLB instantiation. Competence is normalised to a maximum of 1.0. Structural Cost is reported in relative units as the effective number of specialised heads and tools used at the end of the lifetime. The HTF agent is the only one among those we tested that simultaneously maintains high competence and medium structural cost by performing structural triage.

Agent Type	**Final Competence** (max 1.0)	**Final Structural Cost** (heads / tools)	
Plain Global (Flat)	0.65 (stalls in tool-necessity phase)	Low (1 head, 0 tools)	
Flat-Plus (Modular)	0.95 (high)	Very high (12 heads, 3 tools)	E
HTF (Hierarchical)	0.94 (high)	Medium (6 heads, 1 tool)	Structura

1. **Growth phase**: a stream of statistically diverse tasks that encourage adding new specialised experts.
2. **Compression phase**: tasks that are linear or low-rank combinations of previously learned tasks, making many experts redundant.
3. **Tool-necessity phase**: tasks that cannot be accurately solved by the neural backbone alone and require access to an external symbolic tool.

The agents differ only in how they manage structure:

- The *Plain Global* agent maintains a single global head and never changes its architecture.
- The *Flat-Plus* agent can grow new heads when local training loss increases, but it never compresses or removes them.
- The *HTF* agent implements the full structural policy π_s with SPAWN, MERGE, FORGET, and TOOL-ACQUIRE actions driven by the feelings head V_ϕ (MF_c, HR, SEV_{ec}) and optimised for the structural reward $\mathcal{R}_S$ described in Section 4.5.

We track two scalar metrics over the lifetime: (i) *Competence*, defined as task-specific reward (or normalised prediction accuracy) averaged over the most recent tasks, and (ii) *Structural Cost*, defined as

$$\text{Cost} \;=\; w_1{\cdot}(\text{total parameter count}) + w_2{\cdot}(\text{Tool-ACQUIRE count}) + w_3{\cdot}(\text{average FLOPs per step}). \tag{1}$$

The structural reward $\mathcal{R}_S$ then trades off competence and cost as in Section 4.5. For this proof-of-concept, we focus on final-lifetime summary statistics rather than absolute scales.

Table 1 reports representative outcomes after an HCLB lifetime. The numbers are deliberately simple and qualitative, chosen to reflect the behavioural patterns we consistently observe in such toy instantiations.

Figure 1 provides a qualitative view of the dynamics that underpin these summary statistics. The three panels show: (A) competence over time, (B) structural cost over time, and (C) the MF_c feeling trace for the HTF agent.

Panel A illustrates the core competence pattern: the Plain Global agent achieves moderate performance in the growth and compression phases but stalls at around 0.65 once the tool-necessity tasks dominate, as the backbone alone cannot solve them. The Flat-Plus agent, by continually spawning new experts and tools, climbs to high competence (≈ 0.95) even in the tool-necessity phase, but only at the cost of a very large and unwieldy structure. The HTF agent reaches a similar level of competence (≈ 0.94) while maintaining a much smaller, triaged architecture.

Panel B shows how this is achieved. The Flat-Plus agent exhibits monotonically increasing structural cost: whenever a new task is difficult, it simply grows the architecture without ever compressing. By contrast, the HTF agent alternates between growth (SPAWN actions in the early regimes) and compression (MERGE and FORGET actions once SEV_{ec} signals redundancy), followed by a modest structural increase when the MF_c signal triggers a TOOL-ACQUIRE event late in the lifetime. This leads to a medium, rather than maximal, final structural cost.

Finally, Panel C shows the MF_c feeling trace for the HTF agent. In the early phases, MF_c remains low or transient: most conflicts can be resolved by SPAWN or MERGE. In the tool-necessity phase, MF_c drifts upward as the agent repeatedly encounters important states where its current structure cannot achieve high reward. Once MF_c passes a learned threshold, the structural policy selects TOOL-ACQUIRE; the subsequent drop in MF_c reflects that the new tool-enabled structure can now resolve the previously irreducible conflict. This is precisely the behavioural signature predicted by the Decoupling of Feeling Principle: slow, temporally integrated feelings guide discrete structural changes that reconcile competence and efficiency.

In summary, this proof-of-concept toy experiment supports the central qualitative claim of this paper: *in our conceptual benchmark, HTF is the only agent among those we tested that consistently maintains both high competence and high structural efficiency over a long, non-stationary lifetime.* While the absolute numbers are not intended as a competitive benchmark, they demonstrate that an explicit hierarchy of feelings and structural actions can implement the desired "structural triage" behaviour in even the simplest HCLB instantiations.

8 Discussion and Future Work

HTF is intentionally modest in scale but ambitious in scope: it is not a new state-of-the-art model, but a candidate architectural principle for long-lived, self-managing agents. Several directions follow from this work.

Scaling beyond toy environments. Our proof-of-concept experiments use a minimal HCLB instantiation. Extending HTF to richer environments (e.g., multi-task RL benchmarks, realistic tool-use settings, or large language model backbones) is an obvious next step. This would require more sophisticated implementations of V_ϕ and π_s, as well as careful engineering of structural cost terms.

Learning feelings. We have treated feelings as learned, low-dimensional statistics derived from world-model representations and structural history. Many design choices remain open: which auxiliary objectives best capture redundancy, novelty, or frustration? How should we aggregate feelings over time? How do we ensure that feelings remain aligned with long-term structural reward rather than overfitting to short-term dynamics?

Interaction with external oversight. In realistic systems, structural decisions may be constrained or guided by external humans (e.g., limiting which tools can be acquired). HTF could serve as an inner layer that proposes structural changes, subject to external approval. Understanding this interaction between internal feelings and external oversight is an important safety and governance question.

Beyond two feeling families. We have emphasised an epistemic/teleological split, but richer taxonomies of feelings may be useful: risk, alignment, social approval, or long-term coherence could each correspond to distinct value channels. HTF provides the scaffolding to incorporate such channels as they become better understood.

9 Conclusion

We have argued that long-lived autonomous agents face a *Structural Dilemma*: they must remain competent on increasingly hard tasks while operating under finite structural budgets. We proposed the *Decoupling of Feeling* principle: resolving this dilemma requires distinct classes of feelings that separate fast, epistemic efficiency pressures from slow, teleological expansion pressures.

The Hierarchical-Temporal Feelings (HTF) architecture instantiates this principle via a world model, a feelings head, a structural policy over discrete actions, and a persistent memory optimised for a structural reward. The Hierarchical Continual Lifetime Benchmark (HCLB) provides a conceptual and practical setting in which to evaluate structural self-management.

In a minimal toy instantiation of HCLB, we showed that, among the agents we tested, HTF uniquely performs structural triage, maintaining high competence with medium structural cost, while simpler baselines either underfit or become structurally obese. These results are preliminary, but they suggest that explicit hierarchical feelings and structural policies can serve as a promising route toward autonomous, self-managing architectures.

Acknowledgements. I would like to thank Deb and Emo for their feedback, encouragement, and inspiration.

References

fig_poc_htf_traces.pdf

Figure 1: Proof-of-concept dynamics in the HCLB toy environment. **(A) Competence**: the Plain Global and Flat-Plus agents plateau in the tool-necessity phase; only HTF exhibits a sharp competence jump after TOOL-ACQUIRE. **(B) Structural Cost**: the Flat-Plus agent accumulates heads and tools monotonically, while HTF shows growth (SPAWN), compression (MERGE/FORGET), and a small increase when TOOL-ACQUIRE is triggered, keeping the structure in a medium regime. **(C) $\mathbf{MF}_c$ trace (HTF only)**: during the tool-necessity phase, MF_c steadily rises as the agent encounters high-importance, high-loss states; once MF_c crosses a threshold, the structural policy selects TOOL-ACQUIRE, after which MF_c drops as the new structure resolves the persistent conflict.