

Rohn: A Layer-Adaptive Autonomous Negotiation Agent for the ANAC 2026 SCML Standard Track

Ryoshin Hatakeyama
Tokyo University of Agriculture and Technology

June 14, 2026

Abstract

This report describes **Rohn**, an autonomous negotiation agent for the ANAC 2026 Supply Chain Management League (SCML) Standard Track [5]. Rohn fixes its supply-chain position (one of three layer classes: L_0 , middle, and last) at initialization, and every negotiation method dispatches to layer-specific logic. Its negotiation skeleton is based on AS0 [2] (winner of the ANAC 2025 Standard Track), and part of its last-layer buying ceiling and price concession is adapted from CautiousStdAgent [3]. On top of these, we design and integrate storage-cost-aware price control (Profit Band, an inventory-clearance engine, and a market-price cap) and risk-aware quantity control (an inventory guard and demand-following procurement). In 998-world simulations, Rohn matches AS0 and PenguinAgent in mean score while achieving the highest stability (smallest variance and loss rate) among the top group.

mization and risk avoidance conflict: buying much reduces the risk of a production stall but raises inventory cost, while buying too little incurs a shortfall penalty. Crucially, which risk dominates depends on the agent's position in the supply chain. Rohn addresses this with a layered dispatch design as an integrating framework, combining the following six mechanisms for price and quantity control to balance profit and stability.

1. Layered dispatch design.
2. Profit Band (a storage-cost-linked price band).
3. Inventory guard (disabled in the last layer).
4. L_0 -layer inventory-clearance engine.
5. Last-layer multi-stage procurement-needs computation.
6. Market-price estimation cap.

1 Introduction

In the SCML [1] environment, each agent buys input products, processes them in a factory, and sells the output to customers, negotiating with many partners simultaneously over price, quantity, and delivery time.

The central difficulty is that profit maxi-

2 Related Work and Inherited Components

Rohn is not designed from scratch: it incorporates techniques from existing strong agents and integrates them within a layered architecture. We distinguish the borrowed and the original components.

Borrowed. From AS0 [2] we inherit the negotiation skeleton: partner-success-rate selection, the basic `smart_price` logic, and the `powerset`-based search over offer combinations. This skeleton (`distribute`, `powerset`, `counter_all`, and the future-step splitting) originates from the SCML tutorial sample agent (`SyncRandomStdAgent`) and is shared with other ANAC entries such as `PenguinAgent` [4]. From `CautiousStdAgent` [3] we adapt the last-layer buying ceiling and the concave price concession over negotiation time (exponent 0.5).

Original. The mechanisms in Sections 3.1–3.6—layered dispatch, Profit Band, inventory guard, the L_0 clearance engine, the last-layer break-even buffer and demand floor, and the market-price cap—have no counterpart in the borrowed sources. In particular, the L_0 clearance engine is a structural redesign that replaces AS0’s `counter_all` (quantity-matching search) with a different algorithm.

3 Design of Rohn

3.1 Layered Dispatch Design

For an n -layer chain, level 0 (`is_first_level`) is the **L_0 layer**; levels 1 to $n - 2$ are the **middle layer**; level $n - 1$ (`is_last_level`) is the **last layer**. Rohn fixes its position once in `init()`, and every method (`propose`, `respond`, `counter_all`) switches its logic accordingly (Table 1).

3.2 Profit Band

In the middle layer (and on the buying side of L_0), Rohn accepts only contracts expected to be profitable. From the `buy_book` (purchase history) and `sell_book` (sales history) it estimates the actual cost and computes two dynamic bounds. Using the average purchase

Table 1: Dominant risk and main strategy per layer.

Layer	Dominant risk	Main strategy
L_0	Excess inventory	Clearance selling
Middle	Inventory & short.	Profit Band, guard
Last	Shortfall	Demand-following

price $\bar{p}_{\text{buy}}$, production cost c_{prod} , storage cost c_{storage} , and remaining steps r , the selling-price floor and buying-price ceiling are

$$p_{\text{sell}}^{\min} = \bar{p}_{\text{buy}} + c_{\text{prod}} - c_{\text{storage}} \cdot r, \quad (1)$$

$$p_{\text{buy}}^{\max} = \bar{p}_{\text{sell}} - c_{\text{prod}}. \quad (2)$$

Equation (1) allows lowering the price by the storage cost that would otherwise accrue from holding inventory; the floor falls when many steps remain and rises near the end. (In L_0 the storage term is scaled to relax the floor further.) Equation (2) is the buying bound required to profit on each unit sold.

3.3 Inventory Guard

To prevent over-purchasing in the middle layer, the allowable inventory bound is $I_{\max} = L \cdot f(\phi)$, where L is the number of production lines, $\phi = t/N$ is the progress, and f decreases with ϕ . When the current inventory exceeds $I_{\max}$, Rohn rejects buying offers or sets the counter quantity to 0. The guard is *disabled in the last layer* (`INVENTORY_GUARD_OFF`), which prioritizes avoiding the shortfall penalty over limiting inventory.

3.4 L_0 -Layer Inventory-Clearance Engine

Because L_0 receives input products exogenously, treating it as a price-seeking negotiator leaves stock unsold and accumulates storage cost. Rohn instead treats L_0 as a *clearance*

agent that sells anything priced above the storage cost, yielding two L_0 -specific logics.

Dedicated counter logic. Discarding the powerset (quantity-matching) search, the agent decides immediately: if an offer's unit price is at least the selling-price floor (Eq. 1) it is accepted (subject to a safe quantity); otherwise a counter at the clearance price $p_{\text{clearance}} = p_{\min} + \rho(p_{\max} - p_{\min})$ is sent, with ρ (clearance_rate) close to 0 so as to propose near the lower bound.

Future-step offers. Beyond selling out the current day, the agent scans all consumers, finds the nearest future step with the largest unreserved quantity, and sends bulk future-step offers there, suppressing future accumulation early.

3.5 Last-Layer Procurement-Needs Computation

Target and required quantity. The agent tracks the exogenous demand $Q_{\text{exo},t}$ and targets $Q_{\text{target}} = \max(Q_{\text{exo},t}, \bar{Q}_{\text{exo}})$, where $\bar{Q}_{\text{exo}}$ is its moving average. A demand floor keeps the production target $Q_{\text{prod}} = \max(Q_{\text{target}}, 0.7L)$ from collapsing on a temporary demand dip. The required quantity adds a buffer and subtracts the input inventory I_{in} , finished-goods inventory I_{out} , and already-contracted supply Q_{supplied} :

$$Q_{\text{need}} = Q_{\text{prod}} + Q_{\text{buf}} - I_{\text{in}} - I_{\text{out}} - Q_{\text{supplied}}. \quad (3)$$

This target is applied consistently across `counter.all`, `needs.at`, and the future-step offers to avoid inconsistencies between methods.

Break-even buffer. Let r^* be the number of remaining steps at which the storage cost and

the shortfall penalty are equal; from measured parameters $r^* \approx 72$. While more than r^* steps remain, the over-inventory penalty dominates and the buffer is 0; below r^* , the shortfall risk dominates and a buffer is added:

$$Q_{\text{buf}} = \begin{cases} L\beta(1 - r/r^*) & r < r^* \\ 0 & r \geq r^* \end{cases} \quad (4)$$

where β is a buffer coefficient. The break-even idea is general, but tailoring the threshold to the measured SCML economic parameters is a contribution of this work.

Buying price band (adapted from CautiousStdAgent [3]). Selling to exogenous customers, the agent sets a buying ceiling from the back-calculated sales price, $p_{\text{buy}}^{\text{ceil}} = R_{\text{exo}}/Q_{\text{exo},t} - c_{\text{prod}} - \delta$ (with R_{exo} the exogenous sales revenue and δ a margin), and concedes concavely over relative negotiation time $t \in [0, 1]$: $p_{\text{buy}}(t) = p_{\min} + (p_{\text{buy}}^{\text{ceil}} - p_{\min})t^{0.5}$. A low early price encourages early agreement; near the deadline it concedes to the break-even line.

3.6 Market-Price Estimation Cap

On the selling side of the middle and L_0 layers, too high a price reduces agreements and accumulates inventory. Rohn records buyers' offer prices and estimates the prevailing price $\hat{p}_{\text{mkt}} = \frac{1}{|\mathcal{O}|} \sum_{o \in \mathcal{O}} p_o$ over the most recent w steps, then uses it as the selling-price upper bound to avoid proposals that deviate from the market level.

4 Evaluation

We ran simulations in the SCML Standard Track with `n.steps` = 100 and `n.processes` = 2–6 (random per world) over 1000 worlds (998 valid; 2 excluded due to generation errors), using a Tukey truncated mean (truncation

0.05). The comparison agents are AS0 [2], CautiousStdAgent [3], PenguinAgent [4], and GreedyStdAgent [1]. The score is normalized (final balance / initial balance, 1.0 = break-even).

Rohn ranks first in mean score, but the differences among the top three (Rohn, PenguinAgent, AS0) are below 0.01, which is not significant given the score variability (std $\approx$ 0.3); Rohn is thus on par with AS0 and PenguinAgent. However, Rohn has the smallest standard deviation (0.278 vs. 0.329 and 0.336) and the lowest fraction of below-initial-capital worlds among the top group (53.7% vs. 55.2% and 55.3%), indicating a stability-oriented agent with a “solid floor and small variance” (CautiousStdAgent is slightly more stable but lower in mean).

5 Discussion

Rohn is on par with AS0 and PenguinAgent in overall score and has the highest stability among the top group. This stability arises not from one mechanism but from their cooperation: the Profit Band and inventory guard suppress loss-making trades and over-purchasing in the middle layer; the L_0 clearance engine disposes of incoming inventory before storage cost accumulates; the last-layer demand-following procurement suppresses the shortfall penalty; and the market-price cap keeps selling prices near the market level. By addressing each layer’s dominant risk simultaneously, these mechanisms prevent floor collapse across a wide range of worlds, with the layered dispatch design switching them per layer. Rohn’s strength thus lies in combining these original mechanisms on top of the borrowed skeleton and aligning them with the risk structure.

Limitations remain: the upside in booming worlds is somewhat below the top group, and the parameters (clearance_rate, r^* , the mini-

mum safety buffer, etc.) are set empirically and may be suboptimal under environmental change. Future work includes strengthening the booming-world upside and optimizing parameters via reinforcement learning.

6 Conclusion

We presented Rohn, an autonomous negotiation agent for the ANAC 2026 SCML Standard Track. Building on AS0’s [2] skeleton and CautiousStdAgent’s [3] last-layer technique, Rohn combines three groups of original mechanisms within a layered architecture: (i) the **layered dispatch design**, an integrating framework that switches all negotiation methods by the economic risk of each layer; (ii) **storage-cost-aware price control** (Profit Band, the L_0 clearance engine, and the market-price cap), preventing loss-making trades and inventory stagnation; and (iii) **risk-aware quantity control** (the inventory guard and the last-layer demand-following procurement with a break-even buffer $r^* = 72$ and a demand floor). Their cooperation yields performance on par with strong competitors and the highest stability among the top group. Future work includes strengthening the booming-world upside, estimating r^* automatically, and optimizing parameters via reinforcement learning.

References

- [1] Y. Mohammad et al., “SCML: A Framework for the ANAC Supply Chain Management League,” <https://github.com/yasserfarouk/scml>, 2020.
- [2] A. Sadahiro, “AS0 Agent Strategy Report,” ANAC 2025 SCML Standard Track, Tokyo University of Agriculture and Technology, 2025.

Table 2: Score distribution of the agents (normalized score, 998 worlds).

Agent	Mean	Std	Min	25%	Median	Max
Rohn	1.026	0.278	−0.327	0.859	0.988	1.694
PenguinAgent	1.024	0.329	−0.349	0.815	0.984	1.786
AS0	1.019	0.336	−0.868	0.812	0.986	1.852
CautiousStdAgent	0.989	0.264	−0.439	0.899	1.000	1.636
GreedyStdAgent	0.483	0.491	−1.262	0.187	0.606	1.384

- [3] R. Miyajima, “CautiousStdAgent: An agent submitted to the ANAC 2024 SCML Standard Track,” Tokyo University of Agriculture and Technology, 2024.
- [4] K. Goh and T. Otsuka, “PenguinAgent: An agent submitted to the ANAC 2024 SCM league,” Nagoya Institute of Technology, 2024.
- [5] ANAC Organizing Committee, “ANAC 2026: Automated Negotiating Agents Competition,” <https://web.tuat.ac.jp/~katfujii/ANAC2026/>, 2026.