Large Language Models as Decision-Making Agents in Energy Systems

The Case of an Agentic AI Home Energy Management System

Reda El Makroum, Sebastian Zwickl-Bernhard, Lukas Kranzl, Hans Auer

Energy Economics Group (EEG), Technische Universität Wien

elmakroum@eeg.tuwien.ac.at

LLMs Are Reshaping the Energy Landscape

AI-driven workloads are the fastest-growing source of **electricity demand**: global data centers are projected to consume over **1,000 TWh by 2026**.[1]

- AI training and inference are driving **unprecedented load growth**.[2]
- Data center electricity demand projected to **double by 2030**.[1]
- The grid must now plan around data centers as major demand sources.

Energy systems are already adapting to AI as a **new category of demand**.

Utilities, grid operators, and policymakers are actively responding to this reality.

---

[1] IEA, *Energy and AI*, 2025

[2] Jiang et al., "Preventing the Immense Increase in the Life-Cycle Energy and Carbon Footprints of LLM-Powered Intelligent Chatbots," *Engineering*, 2024

What Does This LLM-Driven Demand Look Like?

Two distinct workloads with very different grid profiles:

- **Training**: massive, sustained GPU loads running for weeks or months.
- **Inference**: intermittent, user-driven demand that scales with adoption.

But this demand has a unique property: it is inherently **flexible**.

- Training jobs can be **scheduled and shifted** to off-peak hours or renewable windows.[1]
- Data centers can locate where energy is cheapest and cleanest.

But how else will LLMs change how we deal with energy?

[1] d'Orgeval et al., "Generative AI Impact Assessment through a Life Cycle Analysis of Multiple Data Center Typologies," *Applied Energy*, 2026

Towards Agentic Grid Management

LLM utility extends far beyond generating text and images. Agentic tool use has already transformed software engineering, and the same paradigm shift is now emerging in energy systems.

Across energy systems, LLMs are already being explored for:

- **Grid operation and planning**: load forecasting, fault detection, and power system analysis.[1]
- **Building energy modeling**: automated generation and simulation of building energy models.[2]
- **Context-aware energy management**: conversational AI agents for smart buildings.[3]

[1] Zhang et al., "Large Language Models Meet Energy Systems," *Applied Energy*, 2026
[2] Liu et al., "Large Language Models for Building Energy Applications," *Building Simulation*, 2025
[3] He and Jazizadeh, "Context-aware LLM-based AI Agents for Human-centered Energy Management Systems in Smart Buildings," *arXiv*, 2025

But Why Do We Even Need LLMs Here?

People are a core requirement for the energy transition to happen.

With **prosumers** widely considered as a pillar to a decarbonized energy supply, enabling and sustaining their **active participation** is critical.[1]

Yet this remains one of the most **persistent challenges** in energy systems.

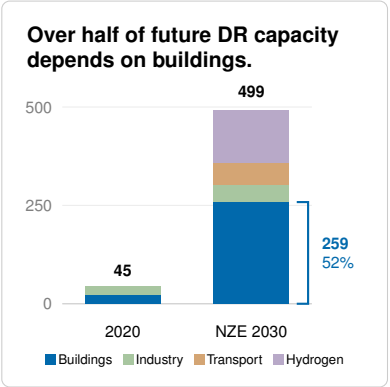


Fig. 1: DR expected capacity growth by sector (GW).[2]

[1] Gunther et al., "Psychological and Contextual Determinants of Clean Energy Technology Adoption," *Nature Reviews Clean Technology*, 2025

[2] IEA, *Energy and AI*, CC BY 4.0, 2025

The Interaction Barrier

Optimizers, markets, and smart devices already exist, but most people **cannot use them** without specialized knowledge.[1]

Prosumers willing to provide flexibility cannot easily understand what is being requested, evaluate whether participation suits them, or verify that compensation is fair.[2]

LLMs can turn **natural language into technical action**: lowering the interaction barrier from expert level to everyday conversation.

---

[1] Vindegg and Julsrud, "Digitised Demand Response in Practice," *Energy Efficiency*, 2024

[2] Stampatori and Rossetto, "From Hesitation to Participation: Examining Behavioural Barriers to Engage Customers in Flexibility Markets," *Current Sustainable/Renewable Energy Reports*, 2024

The Case Study: Home Energy Management

These barriers are particularly evident in **home energy management**.

The residential sector presents the greatest challenge for demand-side flexibility, as participation depends heavily on consumer willingness.[1]

We propose an **agentic AI** approach – where an LLM autonomously selects tools, coordinates actions, and adapts to context – to manage scheduling from natural language requests.

Research question: How can a reliable, secure, and effective agentic AI system for home energy management systems be designed, developed, and simulated?

[1] Sridhar et al., "Toward Residential Flexibility – Consumer Willingness to Enroll Household Loads in Demand Response," *Applied Energy*, 2023

System Architecture of the Agentic AI HEMS

A central orchestrator delegates to three specialist agents, each handling a single appliance type.

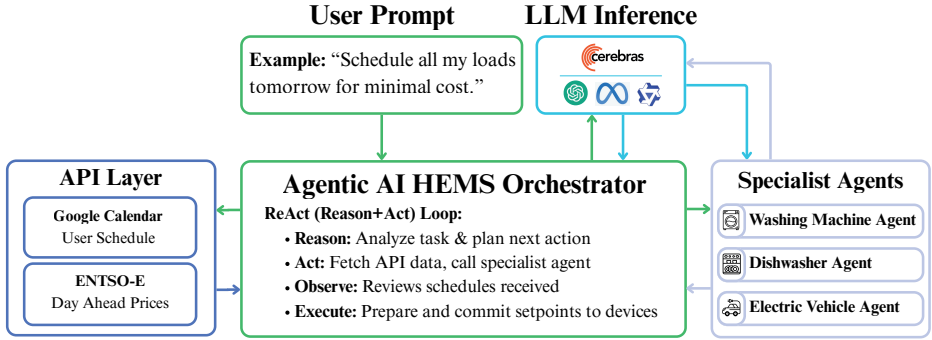


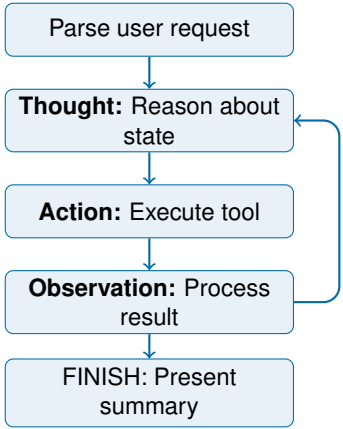
Fig. 2: Agentic AI HEMS architecture with ReAct-based orchestration and specialist agent delegation.

How does the Orchestrator Reason?

The orchestrator follows the **ReAct pattern**[1]: alternating between reasoning about system state and executing actions.

Each iteration produces a **Thought** (what to do next), an **Action** (tool call), and an **Observation** (result), looping until the task is complete.

Specialist agents operate in **single-turn** interactions to minimize token consumption.



[1] Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," *arXiv:2210.03629*, 2023

The Orchestrator Toolset

We develop six tools and make them available to the orchestrator, executed in adaptive sequences based on user requests:

- `get_electricity_prices`: fetches day-ahead prices from ENTSO-E (96 timeslots).
- `get_calendar_constraint`: extracts deadlines from Google Calendar events.
- `calculate_window_sums`: evaluates all valid time windows for cost analysis.
- `call_appliance_agent`: delegates scheduling to a specialist agent.
- `schedule_appliance`: executes the final schedule as a 96-element binary array.
- `finish`: terminates the loop and presents a user-facing summary.

What guides the Orchestrator?

A system prompt defines the agent's role, available tools, and behavioral constraints before any user interaction.

Role: `You are the central coordinator for a Home Energy Management System. Your role is to receive scheduling requests, delegate to specialized appliance agents, and coordinate optimal schedules.`

Available Tools:

1. *`get_electricity_prices(date)`* - Fetches day-ahead prices. Returns: 96 timeslots (EUR/kWh)
2. *`call_appliance_agent(agent_name, prices_data, user_request)`* - Delegates to specialist agent

`[...]`

Required Workflow Order:

1. GET_PRICES (always required)
2. GET_CALENDAR_CONSTRAINT (if EV involved)
3. CALL_AGENT (for each appliance)
4. SCHEDULE (after each recommendation)
5. FINISH (when all schedules executed)

How Do We Test the System?

Three open-source models evaluated via the Cerebras API, a high-speed inference platform for open-source LLMs.

	Single-appliance	Multi-appliance	Analytical queries
Llama-3.3-70B	5 runs	5 runs	15 runs (3 stages)
Qwen-3-32B	5 runs	5 runs	15 runs (3 stages)
GPT-OSS-120B	5 runs	5 runs	15 runs (3 stages)
		75 total runs	

Real Austrian day-ahead prices fetched live from the ENTSO-E API at each run. Temperature 0.0 for deterministic outputs.

What Do We Evaluate Against?

All agent-generated schedules are benchmarked against **MILP optimal solutions** computed via exhaustive search. An agent schedule matching the MILP solution is classified as optimal.

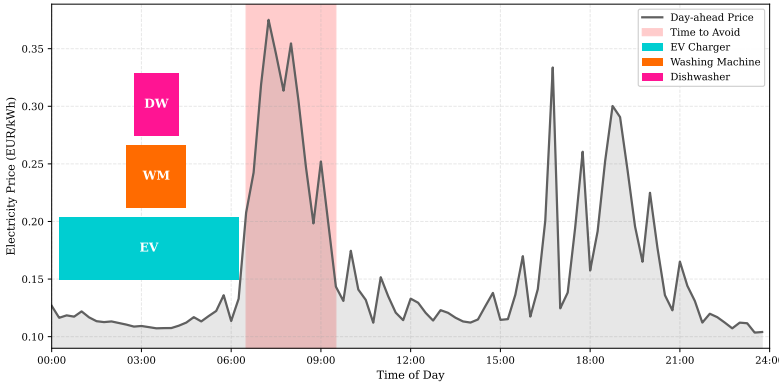
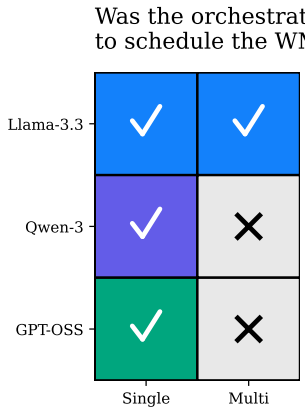
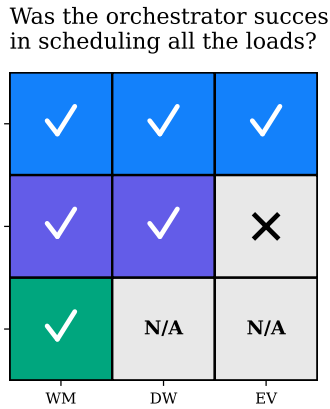


Fig. 3: MILP-optimal multi-appliance scheduling for 15 Oct 2025 using real Austrian day-ahead prices.

How does the Agentic System Perform?



Single-appliance: all models achieve **100% optimality** with similar computational requirements.



Multi-appliance: only Llama-3.3-70B maintains **100% optimality**. Other models fail to coordinate all three appliances.

Why Do Models Fail?

Identical prompt: `"Schedule all my loads tomorrow"`

Llama-3.3-70B (9 iterations)

1. GET_PRICES
2. GET_CALENDAR_CONSTRAINT
3. CALL_AGENT (WM)
4. SCHEDULE (WM)
5. CALL_AGENT (DW)
6. SCHEDULE (DW)
7. CALL_AGENT (EV)
8. SCHEDULE (EV)
9. FINISH (3/3 appliances)

GPT-OSS-120B (5 iterations)

1. GET_PRICES
2. GET_CALENDAR_CONSTRAINT
3. CALL_AGENT (WM)
4. SCHEDULE (WM)
5. **FINISH (1/3 appliances)**

Premature termination
DW and EV skipped

Both models follow identical initial steps, but GPT-OSS-120B **terminates prematurely** after scheduling only the washing machine despite having retrieved EV calendar constraints.

Can the System Handle Analytical Queries?

Beyond scheduling, we test whether the orchestrator can answer **analytical questions** (e.g., “What is the most expensive 3-hour window?”). Three progressive prompt stages assess the minimum guidance required:

Stage 1 – Baseline

// No analytical query guidance provided

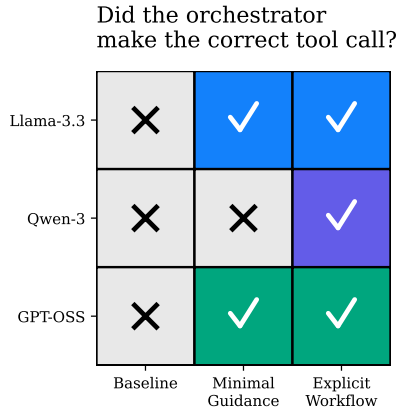
Stage 2 – Minimal Guidance

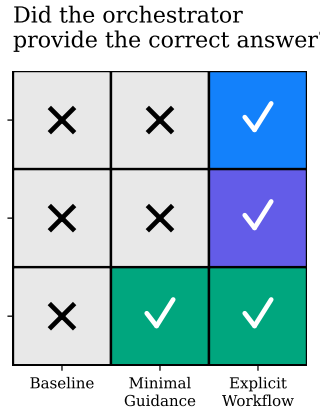
For price analysis queries, use *calculate_window_sums* rather than estimation.

Stage 3 – Explicit Workflow

Analytical Queries: For price analysis, use CALCULATE_WINDOW_SUMS with appropriate window_size (e.g., 1 hour = 4 slots at 15min resolution). To identify expensive periods, use the MAXIMUM sum; to find cheap periods, use the MINIMUM sum.

Analytical Query Results





Baseline: 0% success across all models.

Minimal: GPT-OSS achieves 100%, others fail.

Explicit: **100% success** for all models.

Why Not Just Use a Traditional Optimizer?

MILP solvers compute optimal schedules. But users need technical expertise to specify parameters, constraints, and objective functions.

Adding a natural language interface to an existing optimizer only improves **input accessibility**. The agentic approach enables capabilities that go **beyond scheduling**:

- Answering household energy queries (“How much did I spend this week?”)
- Explaining scheduling decisions in plain language
- Identifying consumption inefficiencies and recommending adjustments
- Adapting to qualitative preferences that resist formalization

These advisory and analytical functions require **reasoning capabilities** that optimization backends cannot provide, regardless of their interface layer.

What Are the Practical Considerations?

Inference infrastructure

Multiple providers now offer high-speed inference (Groq, Together AI, Fireworks AI). Edge deployment on local hardware is increasingly viable.

Prompt and context engineering

Autonomous tool usage is unreliable without explicit guidance. Token minimization reduced inference costs by $\sim$40%.

Energy footprint

LLM inference at scale introduces sustainability trade-offs that traditional HEMS avoid.

Security

Prompt injection, credential extraction, and domain scope validation require multi-layer defense.

Limitations

- **Infrastructure dependency**: standard cloud APIs would increase coordination time from 15 seconds to 4–12 minutes.
- **No thermal loads**: heat pumps and hot water storage depend on building physics rather than user habits, requiring fundamentally different scheduling approaches.
- **No real-world user evaluation**: 75 runs on a single day with one household configuration. A future study is needed to assess how this performs in practice.

Key Takeaways

1. **LLM-based orchestration achieves optimal scheduling** using entirely open-source models. Llama-3.3-70B maintains 100% optimality against MILP benchmarks across all scheduling scenarios in under 15 seconds.
2. **Multi-appliance coordination presents substantially greater reasoning demands** than single-appliance optimization, with only one of three models successfully handling simultaneous three-appliance scheduling.
3. **Analytical capabilities require explicit workflow guidance** in the system prompt. Production deployment must anticipate query types and provide corresponding tool instructions.

Conclusion & Future Directions

Agentic orchestration removes the technical barrier between households and optimal energy scheduling. Where traditional systems require expert configuration, a natural language interface lets any user participate in demand response.

Next steps:

- Real-world pilot deployments to evaluate user acceptance and longitudinal performance.
- Extension to building-level energy management coordinating multiple distributed energy resources.
- Inter-household agentic coordination for peer-to-peer energy trading.

All system components are released as open source to enable reproducibility and further development.

Thank You

Reda El Makroum

Energy Economics Group (EEG), Technische Universität Wien

Email: elmakroum@eeg.tuwien.ac.at

github.com/RedaElMakroum/agentic-ai-hems

TU WIEN TECHNISCHE UNIVERSITÄT WIEN

EEG Energy Economics Group