AmI HMAS: A Hypermedia MAS for Goal-Driven Interactions with Every-day Smart Environments

Alexandru Sorici Victor-Vasile Udriștioiu Cristian Cordoș Andrei Olaru

Department of Computer Science and Engineering, National University of Science and Technology POLITEHNICA Bucharest, Romania

{alexandru.sorici,andrei.olaru}@upb.ro, {victor.udristioiu, ioan.cordos}@stud.acs.upb.ro

Example

Me: "Open the blinds!"

LLM: "Which ones?"

Me: "The ones in the study room."

LLM: "I will open the blinds at 50% in the study room."

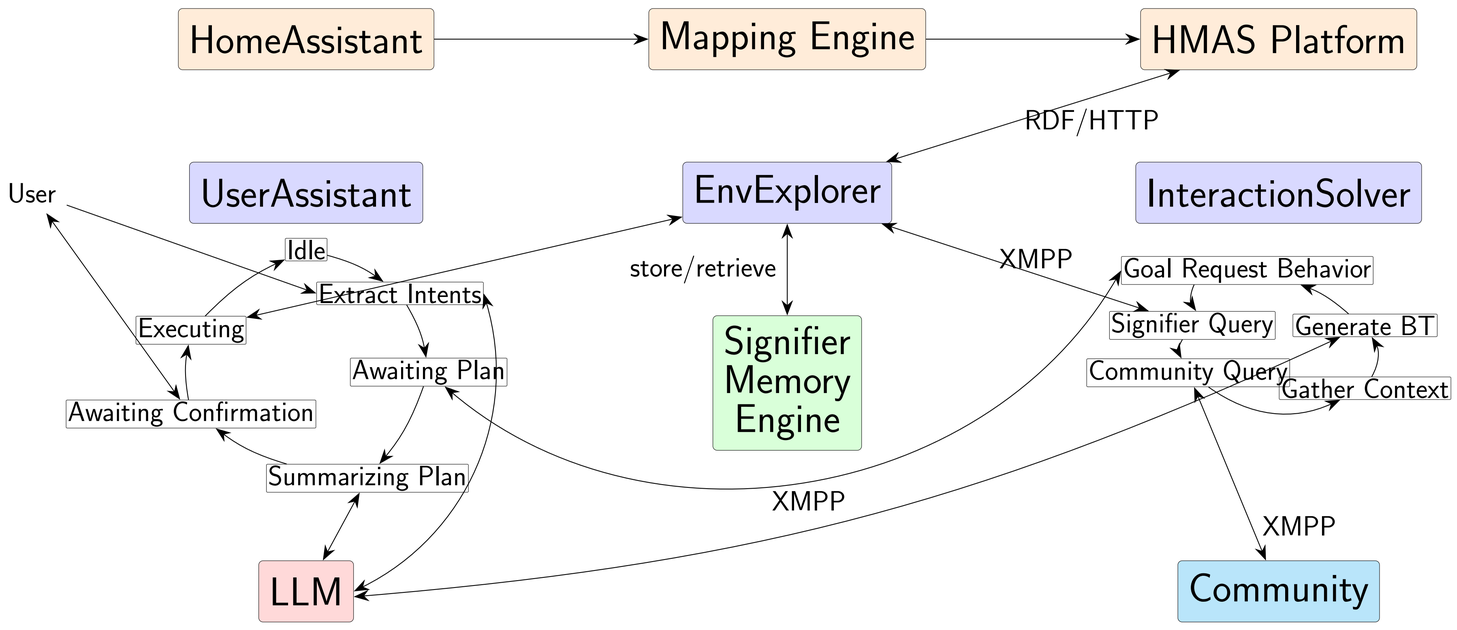
Me: "It's still kind of dark in here..."

LLM: "I will fully open the blinds and turn on the light in the study room."

Me: "Set the blinds to 25% whenever the lights are fully on."

Objective

- Build a framework for goal-driven, LLM-supported interactions in smart enviroments, addressing
  - persistent learning
  - standards-based environment representation
  - experience sharing across deployments



Architecture

- **EnvExplorer** — Discovers and maintains knowledge about the smart environment by crawling the HMAS platform, extracting artifact affordances and monitoring state changes. It also manages the **Signifier Memory Engine** for recalling past **affordance usage experiences** and supports other agents with capability checks, state information, and signifier matching.
- **UserAssistant** — Serves as the user-facing agent responsible for understanding natural language requests, **decomposing** them into **atomic sub-goals**, **classifying** request types, and extracting **structured intents** using LLM reasoning over artifact descriptions. It then forwards these intents for planning and manages the resulting interaction flow.
- **InteractionSolver** — Handles reasoning and planning for user requests by generating structured execution **plans modeled as Behavior Trees (BTs)**. It uses signifier information provided through the **EnvExplorer** to identify suitable affordances and determine sequences of ActionAffordance invocations needed to achieve the user's goals.
  - **Signifier fast-path** — Used for explicit sub-goals with `set` actions. When a request matches a previously used affordance, the InteractionSolver directly constructs a Behavior Tree using the corresponding *ActionAffordance* and the parameters extracted from the structured intent.
  - **LLM planning path** — Used for implicit sub-goals or explicit requests involving `modify` actions. The InteractionSolver first gathers context by retrieving affordance suggestions and relevant environment state information from the EnvExplorer and community knowledge sources, then uses this context together with the structured intent for LLM-based BT plan generation.

Hypermedia Environment

```
<http://localhost:8080/.../artifacts/light308>
  a hmas:Artifact ; td:title "Light308" ;
  td:hasActionAffordance [
    a td:ActionAffordance ;
    td:name "turn_on" ;
    td:title "turn_on" ;
    td:hasForm [
      htv:methodName "POST" ;
      hctl:hasTarget <http://localhost:8080/.../light308/turn_on> ;
      hctl:forContentType "application/json" ;
      hctl:hasOperationType td:invokeAction ]];
```

Community-based Experience Sharing

- **Affordance-based communities** — Agents are organized into communities based on shared affordance types (e.g. *lighting* or *humidity control*) and communicate through SPADE/XMPP to exchange signifiers relevant to user intents.
- **Collaborative signifier retrieval** — When no suitable local signifiers are found during planning, the InteractionSolver queries other community agents with the unresolved intents and integrates the returned *signifiers* as *planning hints*.
- **Knowledge sharing and reuse** — After successful execution of a Behavior Tree, discovered signifiers are stored locally and later shared with other agents requesting support for *similar intents*.
- **Intent-level transfer** — Signifiers are matched at the semantic intent level rather than exact affordance URIs, allowing agents to reuse interaction patterns across different environments with heterogeneous devices and affordance representations.

Experiments

Overall performance summary across all 400 test cases per model.

- *Full success*: tests correctly executed.
- *Partial success*: execution was partially verifiable; at least one action matched.
- *Full+Partial*: combined success rate.
- *Impossible detected*: infeasible intents correctly identified.
- Precision (P), recall (R), F1, and duration aggregated over all tests

Model	Full succ.	Partial succ.	Full+ Partial	Imp. det.	P	R	F1	Dur. (s)
GPT-4o	76.8%	19.0%	95.8%	99.3%	**0.952**	0.908	0.929	15.1
GPT-4o-mini	**80.5%**	15.8%	96.3%	99.6%	0.949	0.934	0.941	17.8
GPT-5-mini	63.8%	**32.5%**	**96.3%**	**100%**	**0.952**	**0.954**	**0.953**	16.1
GPT-5-nano	64.0%	24.3%	88.3%	**100%**	0.943	0.811	0.872	**12.1**

Acknowledgements: This paper is supported by the European Union's HORIZON Research and Innovation Programme under grant agreement No 101120657, project ENFIELD (European Lighthouse to Manifest Trustworthy and Green AI).