An Ecosystem for Scalable Symbolic Modeling in Neurosymbolic AI; or *Shapes of Cognition*

Marjorie McShane MARGEMC34@GMAIL.COM
Sergei Nirenburg ZAVEDOMO@GMAIL.COM
Sanjay Oruganti SANJAYOVS.RPI@OUTLOOK.COM
Jesse English DRJESSEENGLISH@GMAIL.COM
Language-Endowed Intelligent Agents Lab, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

Abstract

Abstract. Symbolic AI has a bad reputation. When used alone, it is associated with small, brittle systems in narrow domains; and when incorporated into neurosymbolic architectures, it tends to serve as a minor flourish to fundamentally empirical systems. But it does not need to be this way. Here we present an ecosystem for developing transparent, scalable, neurosymbolic AI that serves agents, developers, system users, and outside stakeholders alike, while staying true to the scientific grounding of cognitive modeling. This ecosystem underlies Language-Endowed Intelligent Agents (LEIAs) developed within the OntoAgent cognitive architecture and the HARMONIC cognitive-robotic one. This paper has different objectives for different audiences. To readers outside of the symbolic modeling community, it explains why symbolic modeling is useful, feasible, and scalable. To readers within the symbolic modeling community, it proposes specific development methodologies that can help us to collectively make our case to a wide variety of stakeholders, with the goal of expanding the footprint of symbolic modeling in neurosymbolic systems to make the latter transparent, explainable and, ultimately, trustworthy.

1. Introduction

The AI community at large views symbolic modeling with skepticism, and this is understandable. In the early days, limitations on processing speed and memory capacity, compounded by the fact that the field wasn't paying sufficient attention to knowledge *content*, meant that symbolic modeling efforts resulted in small and brittle demonstration systems that could not scale. That is how the notion of a "knowledge bottleneck" took hold. The empirical turn of the 1990s was greeted by a collective sigh of relief and the widespread hope that symbolic modeling might be avoided wholesale. Now, decades later, at the crest of excitement over large language models, it is a perfect time to assess what the field at large has lost by not pursuing symbolic modeling in earnest and how we can most effectively correct course.

What we have lost is modeling human-like cognitive functioning in AI systems, centrally, explainability, transparency, reliability, targeted correctability, and the ability of systems to teach and learn like people do—all of which are required in many critical application areas (McShane et al., forthcoming). The way to correct course is through a novel approach to neurosymbolic AI that (a) has a strong symbolic center (an "orchestrator" in Agentic AI terms) supported by well-selected empirical tools and (b) uses novel methodologies to satisfy the needs of a variety of stakeholders whose buy-in is essential.

Our program of R&D, which pursues these aims, is called Language-Endowed Intelligent Agents (LEIAs). LEIAs are neurosymbolic, multimodal, cognitive-robotic systems implemented in the

HARMONIC architecture, shown in Fig. 1 (Oruganti et al. 2024a, 2024b). The cognitive (strategic) layer primarily relies on knowledge-based methods to support reliability and transparency, whereas the robotic (tactical) layer primarily relies on machine learning, which is effective and sufficient since the associated capabilities (e.g., the "how" of moving a robotic arm) need not be explained. The components of the cognitive and robotic layers function both independently and interactively.

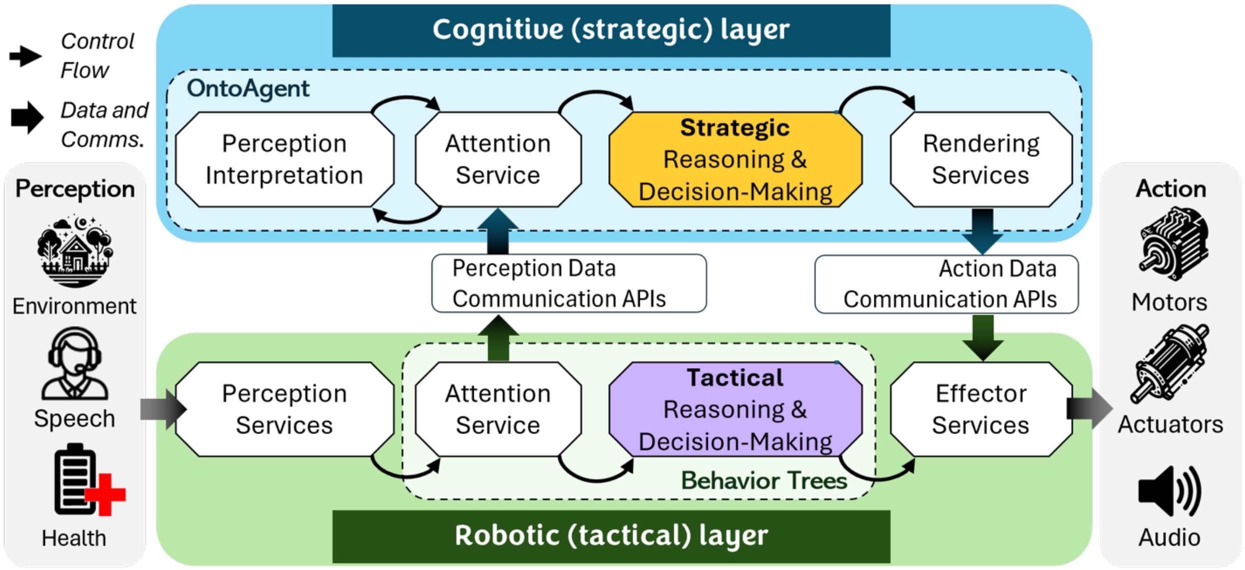


Fig. 1. The HARMONIC cognitive-robotic architecture. The diagram illustrates the processing modules of the architecture. The all-important knowledge substrate of the agent/robot (its ontological world model that includes models of self and other agents, lexicons supporting language understanding and generation, and episodic memory) is omitted for purposes of diagram clarity and is explained in the text.

LEIAs represent a non-traditional take on Agentic AI, which we call OntoAgentic AI. Whereas typical agentic AI systems use LLMs both as the orchestrator and for support functions, OntoAgentic AI uses a LEIA as the orchestrator and leverages both LEIA agents and LLM-based systems for support functions. OntoAgentic AI, therefore, offers reliable, explainable control of overall system operation as well as the cognitive operation of each individual LEIA. This program of R&D is detailed in two recent books that are available open access: *Linguistics for the Age of AI* (McShane & Nirenburg, 2021) and *Agents in the Long Game of AI: Computational cognitive modeling for trustworthy, hybrid AI* (McShane, Nirenburg, & English, 2024). They will be referred to hereafter as LingAI and LongGame, respectively.

This paper describes an ecosystem for developing scalable neurosymbolic AI that serves agents, developers, system users, and outside stakeholders alike, while staying true to the scientific principles and objectives of cognitive modeling. The impetus for a writing paper of this kind is the following: Although it is widely acknowledged that empirical methods are not a full answer to AI (Marcus, 2025), symbolic modeling, if used at all in today's AI systems, still continues to play a subsidiary role, even in most neurosymbolic architectures. We believe that the most promising way to reawaken the larger AI community's interest in symbolic modeling is to build what we call OntoAgentic AI systems using methodologies that make it clear to all stakeholders that the enterprise is scientifically justified, useful, feasible, scalable, and worthy of support.

The paper is intended for two different kinds of readers. To readers outside of the symbolic modeling community, it explains why modern-day symbolic modeling is useful, feasible, and scalable—quite different from its outdated reputation. To readers within the symbolic modeling community, this paper proposes specific development methodologies that can help us to collectively make our case to a wide variety of stakeholders, with the goal of expanding the footprint of symbolic modeling in neurosymbolic AI. No doubt other cognitive systems developers attending this conference build systems with some of the features listed above. However, what we have not seen outside of our team is collecting all of these features in a single package that promises to fundamentally improve the reputation of symbolic modeling. A central requirement is directly addressing knowledge content, to counterbalance the predominance of attention to architectures in the cognitive systems literature (see Nirenburg, McShane & English, 2023, for discussion).

The original version of this paper referred to this ecosystem as *shapes of cognition*, since we find *shapes* to be a useful inspiration in our day-to-day work. In fact, we internally refer to microtheories using names like *shapes of meaning, shapes of coreference, shapes of control*, and so on. Whereas *shapes* might seem like an obvious knowledge engineering anchor for readers schooled in linguistics and/or early AI, for those fully ensconced in statistical AI, the only shapes that might come to mind are nets and black boxes—the latter being *anti*-shapes. If nets and black boxes were enough, then the work of symbolic cognitive modeling would be strictly academic—not uninteresting, but pursued exclusively to satisfy human curiosity. However, nets and black boxes are demonstrably not enough, with contributions arguing this point being too numerous to list (e.g., Connell & Lynott, 2024; Cuskley, Woods and Flaherty, 2024; Kapoor et al., 2024; Xu, Jain & Kankanhalli, 2024). In this version of the paper, we have backed off of the shapes metaphor but suggest that visually oriented thinkers like us might benefit from keeping it in mind throughout.

2. Principles of the LEIA Ecosystem

Below are select principles of the LEIA ecosystem that serve to make its symbolic side scientifically sound, feasible, scalable, explainable, and worthy of support by the broader community. Readers interested in a more exhaustive listing of theoretical and methodological principles can consult LingAI and LongGame.

Human knowledge and reasoning are modeled using formal structures, following a long scientific tradition that spans many fields. Below is a sampling of format-related choices related to ontology, lexicon, and language-oriented reasoning in the LEIA ecosystem:

- Ontology is modeled as a frame-based inheritance hierarchy (not, for example, as the Cyc ontology's "sea of assertions"[1]).
- The same ontological metalanguage is used for episodic memory, the semantic descriptions of lexical senses, and the meaning representations over which agent cognitive processes operate.
- The lexicon structurally aligns syntactic and semantic descriptions of words and constructions.
- Syntactic analysis uses tree structures and transformations that manipulate them.
- The linguistic effects of parallelism—be it syntactic, semantic, lexical, or morphological—are formalized and inform many aspects of language processing.
- Ontological scripts are ontological concepts (using extended expressive means) that predict how the world typically works and guide all aspects of agent operation.
- The more complex scripts are developed and documented using flow charts that are inspired (but not constrained) by best practices of UML code documentation.

Procedural knowledge is grounded in conceptual knowledge, not only in theory but also in practice. Specifically, the code that allows LEIAs to carry out mental and physical actions is organized as procedural attachments to knowledge structures, which not only organizes the codebase (in service of scaling up) but also allows LEIAs to understand and explain their own and others' actions. Informed by decisions on the

[1] Although the Cyc ontology originally used a frame-like architecture, the knowledge representation strategy shifted to a "sea of logical assertions," with each assertion being equally about each of the terms used in it (Mahesh et al., 1996, p. 21). For further discussion, including why we don't use Cyc, see *LongGame* section 3.1.

robotics side of LEIAs, some actions are treated as non-decomposable monoliths, whereas others are decomposable into structured control primitives. Parameterized control schemas encode reference values, control parameters, and error signal quantifications, which supports both feedback and feedforward mechanisms. These modular units enable the system to dynamically compose, tune, and stabilize behaviors in response to human requests, intent, and environmental context, forming a critical interface between high-level reasoning and low-level robotic control and actuation in the HARMONIC architecture.

LEIAs are modeled to get by in a complex world the same way as people do, by orienting around what is typical using generalized recovery methods for what isn't.[2] LEIAs' ability to recognize and reuse complex instances, rather than always engage in first-principles reasoning, reduces the computational burden and increases explanatory power (Gentner, 1983; Newell & Simon, 1972; Thalmann, Souza, & Oberauer, 2019). Associated strategies include acting by habit, reasoning by analogy, satisficing, assessing actionability, and relying on social support from one's team members.

LEIAs are modeled to learn over time in a similar way as people do, which is key to overcoming the knowledge bottleneck and scaling up systems over time. Automatic learning is feasible thanks to (a) the highly structured knowledge bases, (b) the availability of metalevel scripts that guide agents through the learning process, and (c) the availability of large language models that serve as sources of raw data that LEIAs can convert into interpreted knowledge (which must, however, be vetted by people due to the potential for hallucinations). Manual and semi-automatic knowledge acquisition, which are necessary adjuncts to automatic learning, are supported by well-developed methodologies (see LongGame, chapter 9 for details).

Language models are used for what they are good for: manipulating the surface form of language. In addition to the learning-oriented example above, we use language models for various support tasks—such as to select which of multiple semantically correct paraphrases (generated by a LEIA's symbolic processing) is most suitable for the given context, and to help identify knowledge-acquisition priorities, such as the most commonly used words and expressions in task-oriented dialogs across domains. What we are not doing is spending excessive resources on testing the limits of reliability of language models, which is being rigorously explored by others.

The methodology of developing LEIAs supports flesh-and-blood human developers. At the same time as we are explicitly modeling LEIAs, we are also implicitly modeling ourselves as system developers, taking into consideration what can be expected of real people in the day-to-day work of knowledge and system engineering. This explains our prioritization of graphics and tightly organized knowledge and code bases. Whereas agents can operate using data and using code that most humans find impenetrable (as has been amply shown by the history of quick ramp-up demonstration systems), if we are to keep people in the loop as symbolic systems scale up, organization *as assessed from the human perspective* is of utmost importance.

For symbolic modeling to be embraced more widely, a large variety of stakeholders need to buy in. One key to this, we think, is to make real-time agent operation traceable using human-interpretable under-the-hood panels. Developers of symbolic systems—ourselves included—have to face the fact that using proof-of-concept demo systems as a yardstick for progress can fail to impress. Observers are justified in wondering what's underneath, how much of the code will be thrown away in the next round, and whether the approach has any potential for scaling up. Showing what is happening underneath in human-interpretable ways is not difficult if the knowledge bases and algorithms are actually as transparent and explainable as the overarching principles of symbolic modeling would expect them to be.

Ideally, a paper about structures, shapes, and visualizations would be chock full of diagrams. Although this will be possible in the conference talk, it is not possible given space constraints here. Our goals here

[2] Prioritizing frequent phenomena is not the same as the low-hanging-fruit approaches that hamstrung real progress in natural language process. For discussion, see LingAI.

must, therefore, be correspondingly modest: to provide a glimpse into LEIA modeling in support of the generalizations above. In what follows, we describe how we realize our shapes-inspired approach to neurosymbolic AI using three points of departure: the ontology, the lexicon, and applications. The phenomena discussed in each subsection are listed in boldface. It might be useful for readers to skim through them first, to understand the flow of the description. As a final note about scope, this is not a survey paper and shares none of the objectives of that genre.

2. Starting from the Ontology…

LEIA cognition and operation orient around meaning, which is defined in terms of an unambiguous, language-independent ontology, following the theory of Ontological Semantics (Nirenburg & Raskin, 2004).

The ontology is structured as a frame-based, property-rich inheritance network.[3] The power of meaning specification is enhanced by allowing different strengths of constraints on properties, recorded using facets: *value, default, sem*, and *relaxable-to* (distinguished below using differences in typeface weights). A small excerpt from the concept SURGERY illustrates the structure of concept descriptions.

SURGERY		
IS-A	**value**	**MEDICAL-PROCEDURE**
AGENT	**default**	**SURGEON**
	sem	PHYSICIAN
	relaxable-to	HUMAN, ROBOT
LOCATION	**default**	**OPERATING-ROOM**
	sem	MEDICAL-BUILDING
	relaxable-to	PLACE

Organizing the ontology as a structured inheritance hierarchy offers multiple advantages. It allows agents to reason about subclasses and superclasses. It fosters both manual knowledge acquisition and agent learning, since only locally distinct property values of a new concept need be specified once its parent has been identified. It facilitates agent reasoning about the salient distinctions between proximate concepts in the ontological space. And it facilitates broadscale knowledge acquisition by making clear which properties are most salient within a given subhierarchy and, therefore, must be locally specified in all concepts.

Procedural knowledge is recorded as ontological scripts (cf. Schank & Abelson, 1977). Scripts are ontological event frames enhanced by additional expressive means, such as coindexed ontological instances. They record expectations about how complex events typically play out. In applications, they are instantiated as plans, which reflect a particular path through the script with particular participants and props. As an example, below is a small excerpt from the script for filling a gas tank, which shows the top level and the first subevent (in the full script, each subevent heads its own frame, drilling down any number of levels to leaf events).

[3] Related phenomena in the literature include prototypes (Rosch, 1973), memory organization packets (Schank, 1982), and templates (Sung et al., 2021).

```
FILL-GAS-TANK
    IS-A                  MACHINE-MAINTENANCE
    AGENT                 HUMAN-OR-AGENT-#1
    THEME                 GAS-TANK-#1
    CAUSED-BY             FLUID-LEVEL-#1 (DOMAIN GAS-TANK) (RANGE < .2)     ; fuel level is low
    EFFECT                FLUID-LEVEL-#2 (DOMAIN GAS-TANK) (RANGE > .9)     ; fuel tank is full
    HAS-EVENT-AS-PART     REMOVE-#1, INSERT-#1, PUMP-LIQUID-#1, REMOVE-#2, MOVE-#2, CLOSE-CONTAINER-#1
REMOVE-#1
    AGENT                 HUMAN-OR-AGENT-#1
    THEME                 GAS-CAP-#1 (PART-OF-OBJECT GAS-TANK-#1)
    SOURCE                GAS-TANK-#1
    CALL-EFFECTOR         pointer to the robotic code that makes this happen
```

This script says that you fill a gas tank because the gas level is low and the result is that the tank is full. There are six ordered, top-level subevents: open the gas tank, remove the nozzle from the fuel dispenser, insert the nozzle into the gas tank, pump the gas until the tank is full, pull the nozzle out of the gas tank, return it to the fuel dispenser, and close the gas tank. Coreferences among participants and props are indicated using indices that indicate ontological instances. The CALL-EFFECTOR property links the knowledge structure to the code that enables simulated or embodied LEIAs to carry out the action. The fact that CALL-EFFECTOR is appended to REMOVE-#1 indicates that, in the robotic layer of the LEIA, opening the gas tank (i.e., removing the gas cap) is treated as an atomic action that is initiated using a single function call.

Scripts are first developed graphically, using flow charts, and then translated into the ontological metalanguage. In order to develop large, symbolic systems over time without losses, knowledge engineers and system engineers need to sign off on a grain-size of algorithm that stretches both beyond their comfort zones. This is because implementation decisions that might initially seem unimportant can have serious consequences for an algorithm's extensibility (i.e., the system's scalability) over time. This is something we have learned through experience. For example, earlier versions of the language understanding system used by LEIAs started as a quick ramp-up computer engineering project in service of typical NLP applications, such as machine translation and question answering. That system served research goals and prototype applications, but the codebase was accessible only to its developer and proved difficult to transfer to other developers and scale up. Subsequently, we redesigned the language understanding process as an ontological script divided neatly into subscripts that handle different linguistic phenomena, with the leaves of those subscripts calling the associated analytical procedures. This organization of language understanding is detailed in chapters 2-7 of LingAI. The overall top-level algorithm of the entire process is available here: https://homepages.hass.rpi.edu/mcsham2/Appendix-Long-Game/APP-Long-Game-NLU.pdf. Once all human developers in the loop have signed off on a graphic representation of an algorithm, it is reformulated as a set of the ontological scripts that support actual system operation. The graphic representation remains in play for teaching, explaining, and enhancing agent operation.

Scripts can be learned dynamically because both they themselves and the learning process are structured. To overcome the knowledge bottleneck, agents must engage in lifelong large-scale learning. One type of learning that is particularly important for agents serving as apprentices is learning scripts. But script learning is not a single capability. On the one hand, scripts can range from simple to complex; and on the other hand, script learning relies on a large number of enabling capabilities. So, saying that an agent can learn scripts is too vague to be informative. The question is, what *exactly* does this learning involve? Answering this requires decomposing script learning into its many component tasks and, for each one, identifying which actual phenomena the agent can handle at any given time. Our model for doing that is a good example of graphics support not only the modeling itself but also its dissemination.

The top level of the script-learning script is shown in Fig. 2. The gradient coloring is intended to convey that, in a given learning scenario, individual modules can present different development challenge levels, with white being simple and dark blue being difficult.

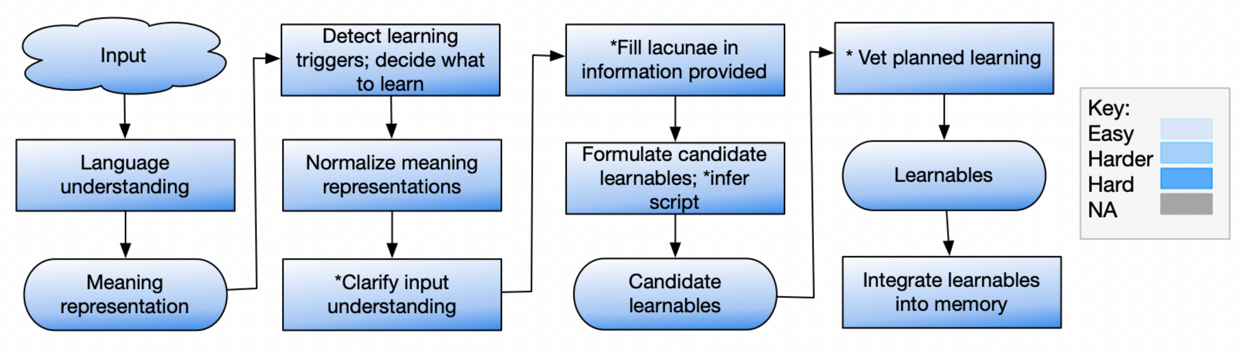

Input
Language understanding
Meaning representation
Detect learning triggers; decide what to learn
Normalize meaning representations
*Clarify input understanding
*Fill lacunae in information provided
Formulate candidate learnables; *infer script
Candidate learnables
* Vet planned learning
Learnables
Integrate learnables into memory
Key:
Easy
Harder
Hard
NA

Fig. 2. The script for learning scripts. Shading reflects varying challenge levels. Asterisks indicate optional modules.

Fig. 3 shows how the coloring strategy captures how different learning scenarios can pose different challenge levels in different modules.

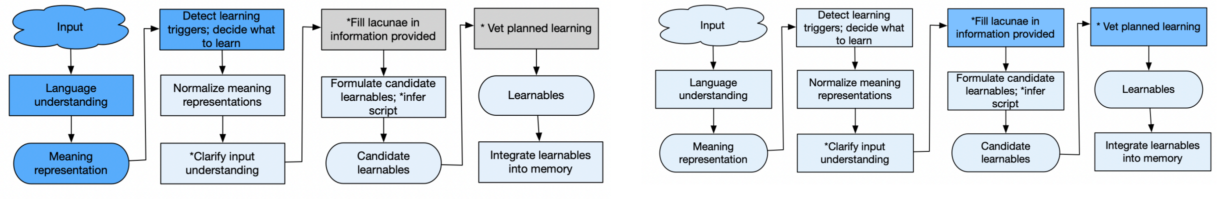

Input
Language understanding
Meaning representation
Detect learning triggers; decide what to learn
Normalize meaning representations
*Clarify input understanding
*Fill lacunae in information provided
Formulate candidate learnables; *infer script
Candidate learnables
* Vet planned learning
Learnables
Integrate learnables into memory
Input
Language understanding
Meaning representation
Detect learning triggers; decide what to learn
Normalize meaning representations
*Clarify input understanding
*Fill lacunae in information provided
Formulate candidate learnables; *infer script
Candidate learnables
* Vet planned learning
Learnables
Integrate learnables into memory

Fig. 3. Color coding helps to visualize different foci and challenges across script-learning scenarios.

In the left-hand scenario, agents receive language inputs that are difficult to understand: they might be highly fragmentary, elliptical, or ambiguous; they might include many unknown words or concepts; and/or they might be structurally complex. But in that scenario, agents are not asked to identify missing information in the script, and they are not asked to doublecheck what they have learned with their human partners. By contrast, in the right-hand scenario, everything is easy except that the agent *is* supposed to detect and fill lacuna as well as describe back what it learned—both of which, for whatever reason, are difficult in the given situation. So, when agents are being evaluated for their ability to learn scripts, they are actually being evaluated for different things based on which challenges the scenario presents. Moreover, when we are training agents to become better learners, the best strategy is to thoughtfully select a subset of challenges to focus on at a time.

Consider some examples of different challenge levels in the different modules, which reflects how closely what the agent encounters matches what it expects (i.e., what is covered by its knowledge bases and reasoning capabilities).

1. Detecting what needs to be learned:
    a. Easy: “Here’s how you [do something]. First [do A]. Then [do B]. And finally [do C].”
    b. A little harder: “[Doing X] requires [doing Y]. But first you have to make sure that [not Z].”
    c. Hard: A cognitive robot is instructed to shadow a person and figure out what needs to be learned, including what constitutes a typical sequence of actions.

2. Clarifying input understanding through dialog:
    a. Easy: The meaning of a single word or the referent for a single referring expression is unclear in an otherwise understood input.
    b. Harder: Multiple words, referring expressions, or speech acts are unclear, requiring the agent to figure out the most efficient strategy for clarifying them (e.g., cycling through them or asking for a paraphrase of the whole thing).
    c. Hard: Multiple aspects of an input are unclear but the agent is authorized to act as long as it has reached an acceptable threshold of understanding and has identified a usefully actionable chunk of communication. Determining what is actionable, and deciding what to do with the incompletely understood parts, can be quite challenging (McShane et al., 2025).
3. Detecting and filling lacunae in the information provided:
    a. Easy: Agents know that they cannot learn a new concept without knowing its parent (its anchor it in the ontology), so if this information is lacking, it needs to be sought out.
    b. Harder: Events in the script might be presented using language that does not make their ordering clear: for example, "You have to X and Y" can imply *in that order* or allow for either order.
    c. Hard: When people describe a complex event, they can imagine many things to be self-evident, such as that coffee beans need to be ground, and that you have to close the windows of a vehicle before washing it.

These examples should suffice to give an idea of (a) how learning functionalities are graded, (b) how that grading is conceptualized as being closer or farther away from explicitly recorded expectations, and (c) how grading helps us to organize and evaluate development efforts.

Two-action scripts, called scriptlets, guide agent behavior in easily visualizable, traceable ways. A counterproductive strategy for developing symbolic systems is to have idiosyncratic approaches to treating each different phenomenon. A characteristic example is having a dedicated dialog model to handle exactly and only dialog. This is redundant. In LEIAs, dialog is treated like any other action, and the typical *serve-return* expectations about dialog interactions are treated in a generic way using two-action scripts we call scriptlets. The pairs of actions in scriptlets are linked using the property HAS-ADJACENCY-PAIR and its inverse ADJACENCY-PAIR-OF. Table 1 shows a sample subtree of adjacency pairs that involve requests for explanations.

Table 1. The ontological subtrees involving explanation, unexpanded. At all levels, the paired concepts are linked by the relations HAS-ADJACENCY-PAIR and ADJACENCY-PAIR-OF.

- REQUEST-EXPLANATION	- PROVIDE-EXPLANATION
+ REQUEST-INFO-AGENT-PERCEPTION	+ EXPLAIN-AGENT-PERCEPTION
+ REQUEST-INFO-AGENT-ACTION	+ EXPLAIN-AGENT-ACTION
+ REQUEST-INFO-AGENT-KNOWLEDGE	+ EXPLAIN-AGENT-KNOWLEDGE
+ REQUEST-AGENT-REASONING	+ EXPLAIN-AGENT-REASONING

Table 2 shows an expanded subtree whose leaf concepts include links to procedures that detect and respond to each kind of request for explanation. For example, the boldface concepts prepare the agent to detect and respond to questions about what the agent heard, what it thought somebody pointed to, and what it saw.

Table 2. An example of a pair of expanded subtrees.

- REQUEST-INFO-AGENT-PERCEPTION	- EXPLAIN-AGENT-PERCEPTION
- REQUEST-INFO-PERCEPTION-RECOGNITION	- EXPLAIN- PERCEPTION-RECOGNITION
- REQUEST-REPEAT-STRING	**- REPEAT-STRING**
- REQUEST-POINTED-TO-OBJ	**- CONVEY-POINTED-TO-OBJ**
- REQUEST-SEEN-OBJ	**- CONVEY-SEEN-OBJ**

Many kinds of events outside of dialog have adjacency pairs: telling a joke pairs with laughing, waving ‘hi’ to someone pairs with waving ‘hi’ back, and so on. Adjacency pairs, like all ontological knowledge, record how things tend to work in the world. Agents have general procedures that handle situations that do not correlate with the most typical expectations.

The agent’s thoughts are recorded as meaning representations that mirror the structure of ontological knowledge. When LEIAs interpret stimuli, reason, or plan, they do it using the same metalanguage as the ontology. For example, the following meaning representation expresses the idea that *Tony was watching a tiger*.

```
VOLUNTARY-VISUAL-EVENT-1
    AGENT              HUMAN-1
    THEME              TIGER-1
    TIME               < find-anchor-time
    ASPECT             progressive
    episodic-mem       VOLUNTARY-VISUAL-EVENT-#9
HUMAN-1
    HAS-NAME           Tony
    episodic-mem       HUMAN-#17
TIGER-1
    DISCOURSE-STATUS   new
    episodic-mem       TIGER-#1
```

Although it is difficult and expensive to enable agents to interpret their experiences, reason, and learn in terms of an ontological model, there are four main benefits to doing this: (1) interpreted knowledge structures are unambiguous and optimally suited to goal-oriented reasoning, unlike uninterpreted linguistic or non-linguistic data; (2) agents can use all of the knowledge stored about ontological concepts in their reasoning; (3) all of their knowledge and traces of their reasoning are human-inspectable, which will allow humans to develop trust in agent systems; and (4) the vast majority of agent knowledge and reasoning is language-independent, which allows for LEIAs to be ramped up in any natural language with only a change in the language processors at the flanks of the cognitive architecture.

Meaning representations are stored in episodic memory, which mirrors the shape of the ontology and enables agents to reason about instances using knowledge of their types. Episodic knowledge is not part of ontology, but it structurally mirrors the ontology, differing in that it records indexed knowledge about instances rather than types of concepts. The correlation between the ontology and episodic memory means that agents can reason about elements of episodic knowledge by consulting the concept in the ontology, along with its full property-based description. Below is an excerpt from a remembered instance of SURGERY: a particular SURGEON (indexed as #14 in the agent’s memory) operated on a particular PATIENT (#89) in a particular HOSPITAL (#3) on December 12, 2024.

```
SURGERY-#10
    AGENT         SURGEON-#14
    BENEFICIARY   PATIENT-#89
    THEME         APPENDIX.PART-OF.PATIENT-#89
    LOCATION      HOSPITAL-#3
    DATE          2024-12-12
```

Each of the property fillers (SURGEON-#14, PATIENT-#89, HOSPITAL-#3) also has its own property-rich description in episodic memory.

Structured episodic knowledge supports reasoning by analogy. Reasoning by analogy enables agents to circumvent reasoning from first principles by repeating something that worked in the past (e.g., Gentner & Smith, 2013). For example, if an agent needs to create a plan from an option-filled ontological script, the least-effort strategy is to copy the last plan that worked or some other previous plan that worked well or frequently, as long as the given circumstances don't block that plan. Similarly, if the agent receives an ambiguous input, but its past analyses of an identical or similar input resulted in a particular interpretation, then the least-effort action is to interpret the new instance in the same way. For example, if an agent's human collaborator has formerly said "I need a cup of coffee" as a signal that he's about to take a break (not as a request that the agent get him one), then the agent can select that interpretation without extensive reasoning or the need to initiate a clarification dialog.

Of course, matching in service of analogical reasoning can be tricky: *how* similar and *in what ways* do past and current meaning representations (situations, inputs, decisions, etc.) need to be in order to warrant reasoning by analogy? This could spiral into endless complexity but our practical approach to agent modeling helps. The fact is, LEIAs don't need to be able to reason by analogy at a human level in order to use analogy as a useful tool. We can prepare them to reason by analogy in specific ways in specific kinds of situations, depending on user requirements. If they recognize a constellation of feature values (a pattern or shape) that enables them to reliably reason by analogy, then they do. If not, they use some other reasoning strategy. Of course, this requires that we, as developers, specify exactly how we want reasoning by analogy to work, without offloading it to the unreliable operation of language models. Explicit cognitive modeling of this sort contributes both to cognitive science and to the development of reliable agent systems.

Structured episodic knowledge supports memory consolidation. For example, if the agent observes multiple instances in which its human partner, Lou, puts his hammer, screwdriver, or wrench back in his toolbox right after he uses it, then this is best consolidated into the fact that Lou always puts back his tools after he uses them. As with reasoning by analogy, we must give LEIAs specific reasoning procedures for identifying what counts as a habit. These procedures (as yet not developed) will be attached to the concept IDENTIFY-HABIT. There will be a similar concept, CREATE-HABIT, whose functional attachment guides the agent in morphing its own deliberative action (involving System 2 reasoning) into a habitual one (involving System 1 reasoning).[4]

Structured episodic knowledge supports plan selection, since an individual's past preference for how to carry out an action can be copied when creating a new plan from an option-filled script. For example, two different people might teach the agent to carry out a complex action in two different ways, as we demonstrated in past proof-of-concept systems. The agent records both of these as options in its ontological script for the action, and it records the actual preferences of each teacher in its episodic memory of those teaching scenarios. Effectively, these memories are like different stencils over the full script, providing guidance for how the agent should create its plan when working with each collaborator.

3. Starting from the Lexicon…

The computational lexicon used by LEIAs is key to language understanding, language generation, and learning through language.

The lexicon is organized around correlations of structured syntactic and semantic descriptions.[5] Human-oriented lexicons and syntactic theories of language classify word senses according to constructions

[4] See, e.g., Sun, Slusarz & Terry (2005) for more on the two-system view.

[5] For related literature, see Fillmore & Baker (2009) on frames, and Hoffmann & Trousdale (2013) on construction grammar. Note that the construction *semantics* used by LEIAs differs from construction *grammar* in that it is

like transitive, intransitive, and ditransitive. Our approach to lexical storage, called *construction semantics*, takes constructions to a new level by precisely specifying the aligned syntactic and semantic expectations of word senses (see LongGame, sections 3.3 and 4.2.2). As an example, Table 3 shows three lexical senses that have the same semantic structure, recorded in their sem-struc zones (ADMIRE with an AGENT and a THEME), but different syntactic structures, recorded in their syn-struc zones. Variables indicate cross-references, and ^ indicates "the meaning of" the variable. The class names for the syntactic and semantic structures are values of the syn-class and sem-shape fields, respectively.

Table 3. Three constructions with the semantic shape (in boldface) but different syntactic shapes.

admire-v1	look-v24	put-v29
ex: John admires his uncle.	ex: John looks up to his uncle.	ex: John puts his uncle on a pedestal.
syn-class: v-trans	syn-class: v-part-pp	syn-class: v-do-pp
sem-shape: EVENT(AGENT,THEME)	sem-shape: EVENT(AGENT,THEME)	sem-shape: EVENT(AGENT,THEME)
syn-struc subject $var1 v $var0 directobject $var2 sem-struc **ADMIRE** **AGENT ^$var1** **THEME ^$var2**	syn-struc subject $var1 v $var0 part up pp prep to obj $var2 sem-struc **ADMIRE** **AGENT ^$var1** **THEME ^$var2**	syn-struc subject $var1 v $var0 directobject $var2 pp prep on obj det a n pedestal sem-struc **ADMIRE** **AGENT ^$var1** **THEME ^$var2**

Lexical senses can include procedural attachments, many of which rely on predictive structural features of language. We could cite innumerable examples of this since language is a very organized system, but for reasons of space we constrain ourselves to three. (1) When modifiers like *very* and *extremely* are used to modify scalar attributes, they pull their values up or down the relevant scale in ways that can be approximated using a simple model: the meaning of *pretty* is recorded as AESTHETIC-ATTRIBUTE .8 (on the abstract scale {0,1}); *very pretty* pulls it up to .9, and *extremely pretty* to 1. (2) The adverb *respectively* triggers a predictable procedure that structurally realigns compared entities: *Tom and Edith drive a Porsche and a BMW, respectively* remaps to *Tom drives a Porsche and Edith drives a BMW*. (3) Referring expressions—even challenging ones like elided ones and those with broad reference—can be resolved with high confidence when they occur in coreference-predicting configurations. For example, even though the pronoun *it* can be difficult to resolve in many contexts, it is easy to resolve with high confidence when identical or feature-matching direct objects occur in sequential, coordinated verb phrases: *Patty grabbed the cupcake and scarfed it down; Patty grabbed it and scarfed it down*. These examples give only the smallest taste of how formalizable patterns of language use can be implemented as algorithms attached to lexical senses. This not only organizes the knowledge underlying language processing, it also allow for the individual algorithms treating different phenomena to be enhanced as resources permit. For further discussion of coreference, including explanatory flowcharts, see chapter 5 of LingAI and chapter 5 of LongGame.

One of the steps in language understanding is aligning the shapes of syntactic parses with the shapes of the syntactic descriptions of lexical senses. Language understanding by LEIAs is a six-stage

computational rather than theoretical, it treats semantics as centrally as syntax, and it grounds meaning in a formal ontology.

process that involves multiple stages of syntactic analysis followed by multiple stages of semantic and pragmatic analysis, as detailed in LingAI and LongGame. The stages that involve syntax are Basic Syntax, which involves an external parser, and two sequential stages of LEIA processing: OntoSyntax and Basic Semantics. Continuing with our example *Tony was watching a tiger,* OntoSyntax determines that the first verbal sense of watch, watch-v1 (Table 4) is syntactically compatible with the syntactic parse (both are transitive), and Basic Semantics determines that this sense is semantically compatible with the input (*Tony* is an ANIMAL and *a tiger* is a PHYSICAL-OBJECT).[6]

Table 4. The lexical sense *watch-v1* can be used to analyze *Tony was watching a tiger,* resulting in the meaning representation to the right. The shape of the sem-struc in *watch-v1* copies into the meaning representation, as shown using boldface.

watch-v1 syn-struc subject $var1 v $var0 directobject $var2 sem-struc **VOLUNTARY-VISUAL-EVENT** **AGENT** ^$var1 (sem ANIMAL) **THEME** ^$var2 (sem PHYSICAL-OBJECT)	**VOLUNTARY-VISUAL-EVENT-#9** **AGENT** HUMAN-#17 **THEME** TIGER-#1 TIME < find-anchor-time ASPECT progressive lex-map watch-v1

However, not all examples are so simple because word senses can be used in non-basic ways as well. For example, the sentence *Mary needed to feed Spot before going out to dinner* includes three verbs whose basic forms are recorded in the lexicon as shown in table 5.

Table 5. Three lexical senses needed to understand *Mary needed to feed Spot before going out to dinner.*

need-v2 def need plus an xcomp ex I needed to do my homework syn-struc subject $var1 verb $var0 xcomp $var2 sem-struc MODALITY TYPE OBLIGATIVE VALUE 1 SCOPE ^$var2 ATTRIBUTED-TO ^$var1 ^$var2 AGENT ^$var1	feed-v1 def To give food to; transitive ex She fed the dog syn-struc subject $var1 verb $var0 directobject $var2 sem-struc FEED AGENT ^$var1 BENEFICIARY ^$var2	go-v54 def phrasal: X go out to dinner ex We're going out to dinner. syn-struc subject. $var1 verb $var0 prep-part out pp prep to obj dinner sem-struc EAT-AT-RESTAURANT AGENT ^$var1

In our sentence, the first verb, *need*, is used in its basic form (all of its expected syntactic dependencies are accounted for), but the others, *feed* and *go*, are not. *Feed* needs to be converted into an infinitive clause and its missing subject needs to be understood as coreferential with that of *need;* and *go* needs to be converted into a present participle and its missing subject needs to be understood as coreferential with that of *need and feed.* In the tradition of generative grammar, dynamic modifications to recorded lexical knowledge are treated using transformations (Chomsky, 1957). But, whereas generative grammar considers only the

[6] The semantic constraints are written in gray because they are not written explicitly in the lexicon; they are drawn from the ontology during language analysis and generation.

syntactic aspect of transformations, LEIAs need to carry along the semantic interpretations as well—a process detailed in LongGame.

To recap, semantic analysis is informed by the syntactic parse, which is in the shape of a tree. That shape must either directly align with the syntactic shapes (syn-strucs) of argument-taking words in the lexicon or dynamic transformations—which map base shapes into derived shapes—must account for the discrepancies. If a particular input is not properly understood by the semantic analyzer, possible fail points are the lack of a lexical sense of the needed shape or the lack of a transformation to convert an existing lexical sense into the shape needed by the input. Knowledge engineers and software engineers jointly determine—through a combination of introspection and testing—how robust dynamic transformations can be, and when it is better to record complex constructions as explicit shapes in the lexicon, thus obviating the need to execute transformations.

Language generation leverages underspecified ontological templates that reflect standard shapes of meaning. Language generation involves the following steps: first the agent recognizes the need to say something; then it formulates the meaning it wants to express using an ontologically-grounded meaning representation; and finally it decides how to express it using the word senses stored in its lexicon (LongGame*,* Section 4.3).[7] Here we describe how the last step relies on ontological structures that we call shapes of meaning.

Shapes of meaning are variable-inclusive templates for ontological concepts that we hypothesize scaffold human thought just as linguistic constructions scaffold human languages. For example, just as the sentence "Who do you think wanted to be selected?" contains nested constructions that are part of a person's knowledge of English (a wh-question scopes over a verb that selects an infinitival complement), the *meaning* of that sentence reflects nested frames of ontological concepts that are independent of any natural language. We computationally model the process of language generation by formalizing the links between shapes of meaning and their corresponding linguistic constructions.[8]

The key insight of *shapes of meaning* is that the overall shape of the meaning to be expressed (i.e., the full proposition) affects how the individual components need to be expressed. Consider the examples in Table 6, all of which include the information that a bicycle is blue. Depending on what else is included in the meaning representation, the bicycle's blueness is expressed variously in English.

Table 6. Different meaning representations and renderings involving a bicycle's blue color, with time and aspect removed for clarity of presentation.

BICYCLE-#2 COLOR blue	COST-#1 DOMAIN BICYCLE-#2 RANGE .8 BICYCLE-#2 COLOR blue	AMUSE-#1 CAUSED-BY COLOR-#1 EXPERIENCER HUMAN-#1 COLOR-1 DOMAIN BICYCLE-#3 RANGE blue
The bicycle is blue.	The blue bicycle is expensive.	The fact that the bicycle was blue amused me.

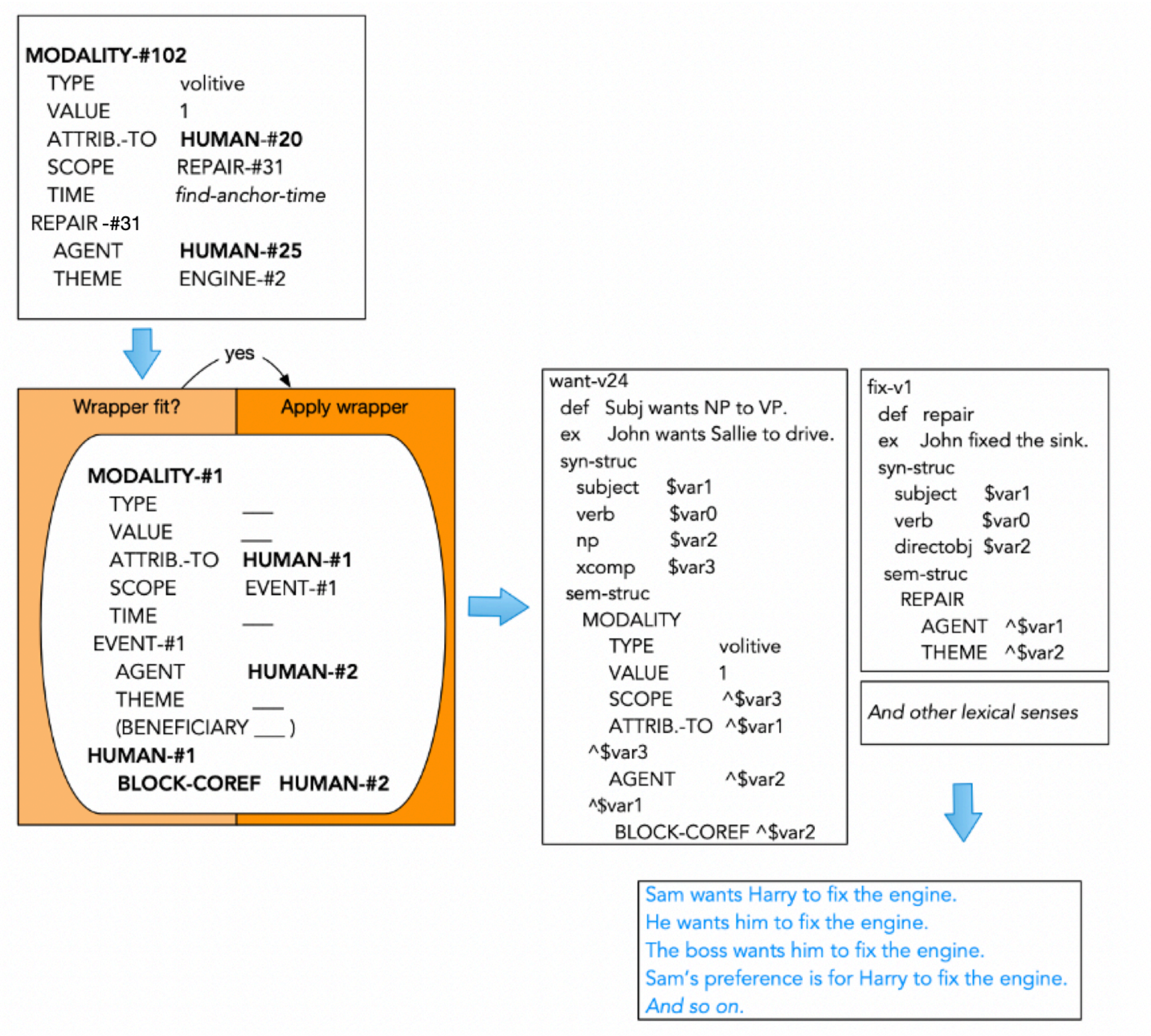
The shape in the left-hand column is an OBJECT described by an ATTRIBUTE. Given that shape in isolation—it is the entire idea to be expressed—the agent needs to create a copular sentence, which is a sentence with the verb *to be*. By contrast, when an OBJECT modified by an ATTRIBUTE is used as a case-role filler in

[7] Actually, the agent can also generate language reflexively, as to shout "Fire!" when it perceives fire, but that goes beyond the scope of this paper.

[8] An obvious question is, why doesn't language understanding require shapes of meaning? As explained in LongGame, language understanding and language generation pose very different challenges to agents that are overcome in different ways.

another frame, as in the middle column, then a prenominal adjectival modifier is the default choice. Finally, when an ATTRIBUTE heads its frame and is used as a case-role filler in another frame, then a formulation like “the fact that N is Adj” is needed.

What we notice here is that the agent must first *detect* the shape of the overall meaning to be expressed and then figure out how to select and manipulate (using transformations) associated lexicon entries to create a sentence. This processing is carried out by two different routines that wrap the shape of meaning, as illustrated in Figure 4.



```
MODALITY-#102
   TYPE          volitive
   VALUE         1
   ATTRIB.-TO    HUMAN-#20
   SCOPE         REPAIR-#31
   TIME          find-anchor-time
 REPAIR -#31
    AGENT        HUMAN-#25
    THEME        ENGINE-#2
```

```
MODALITY-#1
   TYPE           ___
   VALUE          ___
   ATTRIB.-TO     HUMAN-#1
   SCOPE          EVENT-#1
   TIME           ___
 EVENT-#1
    AGENT         HUMAN-#2
    THEME          ___
    (BENEFICIARY ___ )
HUMAN-#1
    BLOCK-COREF   HUMAN-#2
```

```
want-v24
  def   Subj wants NP to VP.
  ex    John wants Sallie to drive.
  syn-struc
    subject    $var1
    verb       $var0
    np         $var2
    xcomp      $var3
   sem-struc
     MODALITY
        TYPE          volitive
        VALUE         1
        SCOPE         ^$var3
        ATTRIB.-TO    ^$var1
     ^$var3
        AGENT         ^$var2
     ^$var1
        BLOCK-COREF ^$var2
```

```
fix-v1
  def   repair
  ex    John fixed the sink.
  syn-struc
     subject     $var1
     verb        $var0
     directobj   $var2
   sem-struc
     REPAIR
        AGENT    ^$var1
        THEME    ^$var2
```

And other lexical senses

Sam wants Harry to fix the engine.
He wants him to fix the engine.
The boss wants him to fix the engine.
Sam’s preference is for Harry to fix the engine.
And so on.

Fig. 4. Visualizing a shape of meaning (in the orange box) being used for language generation.

The meaning representation to the upper left reflects what the agent wants to say. This example assumes that HUMAN-#20 is the LEIA’s memory anchor for a man named Sam who is a boss, and that HUMAN-#25 is the memory anchor for a man named Harry. The relevant shape of meaning is shown in the orange box. It covers the case when one human or intelligent agent wants, needs, requires, etc., another one to do something. Wanting, needing, and requiring are examples of MODALITY: e.g., the meaning *want* is MODALITY (TYPE volitive) (VALUE 1). Modalities scope over EVENTs that, themselves, can be of any shape.

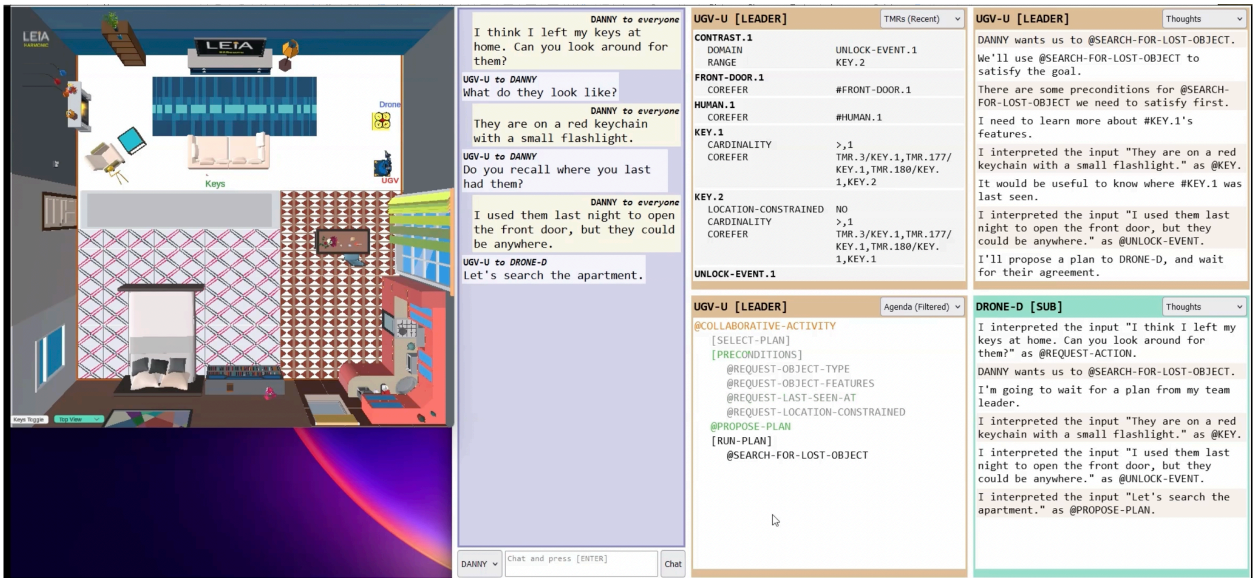
In our example, the EVENT takes an AGENT, a THEME, and an optional BENEFICIARY. The key feature of this shape is that the MODALITY is ATTRIBUTED-TO someone different than the AGENT of the EVENT (i.e., they are not coreferential). So, this shape can be used to generate the sentences shown in Fig. 2 (as well as many more), but it cannot be used to generate *John wants to <has to, must, is trying to* etc.> *fix the engine* since, in these, the modality is attributed to the same individual who is carrying out the event. The routine called "Wrapper fit?" determines whether a given shape is applicable for processing the given meaning representation, and the routine called "Apply wrapper" guides the agent in using the lexicon to express the meaning, which centrally includes modifying the basic, stored information using transformations. The ongoing work of developing *shapes of meaning* involves positing shapes (deciding which frames to include, which are variable and fixed elements, etc.) and testing their combinability and generativity using ever more complex meanings.

Readers familiar with machine learning might wonder why we don't just have the agent learn the correspondences between meaning representations and sentences of English using a large corpus of such pairings, by analogy with machine translation. There are both practical and scientific reasons not to do so. The practical reason is that no such corpus exists and it would require a very large, expensive knowledge acquisition effort to acquire a large enough one. The scientific reason is that, even if the former could be done, it would not contribute to our understanding of human language processing, which is one objective of this program of work.

4. Under the Hood of LEIA Cognition

LEIA systems are deployed in a kit with *under-the-hood* panels that show dynamic, human-interpretable traces of system functioning (LingAI, section 8.1.5; LongGame*,* section 8.7; Nirenburg et al., 2024). We first introduced under-the-hood panels in the Maryland Virtual Patient (MVP) proof-of-concept clinician training application (McShane et al., 2008; McShane & Nirenburg, 2021, ch. 8), where they showed traces of the physiological simulation of the virtual patient, the patient's interoception (perception of bodily sensations), its thoughts, the knowledge it learned, and how it interpreted text inputs from the user, who was playing the role of attending physician. For screen shots, see https://homepages.hass.rpi.edu/mcsham2/Appendix-Materials/Appendix-Ch-8-MVP-Screen-Shots.pdf.

More recently, we have included under-the-hood panels in multiple simulation systems that demonstrate the cognition of robots implemented within our new HARMONIC architecture. Fig. 5 illustrates the use of under-the-hood panels in a demonstration system in which two simulated robots, a drone and a ground vehicle, work as a team to fulfill a search-and-retrieve request by a person. Specifically, a human named Danny, who is located remotely, asks the team to find his keys, which he misplaced in his apartment.



```
DANNY to everyone
I think I left my keys at home. Can you look around for them?
UGV-U to DANNY
What do they look like?
DANNY to everyone
They are on a red keychain with a small flashlight.
UGV-U to DANNY
Do you recall where you last had them?
DANNY to everyone
I used them last night to open the front door, but they could be anywhere.
UGV-U to DRONE-D
Let's search the apartment.
```

```
UGV-U [LEADER]                         TMRs (Recent)
CONTRAST.1
  DOMAIN                   UNLOCK-EVENT.1
  RANGE                    KEY.2
FRONT-DOOR.1
  COREFER                  #FRONT-DOOR.1
HUMAN.1
  COREFER                  #HUMAN.1
KEY.1
  CARDINALITY              >,1
  COREFER                  TMR.3/KEY.1,TMR.177/
                           KEY.1,TMR.180/KEY.
                           1,KEY.2
KEY.2
  LOCATION-CONSTRAINED     NO
  CARDINALITY              >,1
  COREFER                  TMR.3/KEY.1,TMR.177/
                           KEY.1,TMR.180/KEY.
                           1,KEY.1
UNLOCK-EVENT.1
```

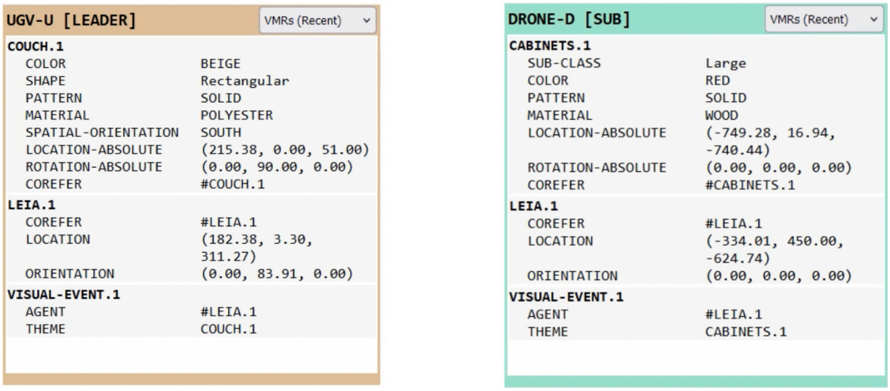
```
UGV-U [LEADER]                         Thoughts
DANNY wants us to @SEARCH-FOR-LOST-OBJECT.
We'll use @SEARCH-FOR-LOST-OBJECT to satisfy the goal.
There are some preconditions for @SEARCH-FOR-LOST-OBJECT we need to satisfy first.
I need to learn more about #KEY.1's features.
I interpreted the input "They are on a red keychain with a small flashlight." as @KEY.
It would be useful to know where #KEY.1 was last seen.
I interpreted the input "I used them last night to open the front door, but they could be anywhere." as @UNLOCK-EVENT.
I'll propose a plan to DRONE-D, and wait for their agreement.
```

```
UGV-U [LEADER]                         Agenda (Filtered)
@COLLABORATIVE-ACTIVITY
  [SELECT-PLAN]
  [PRECONDITIONS]
    @REQUEST-OBJECT-TYPE
    @REQUEST-OBJECT-FEATURES
    @REQUEST-LAST-SEEN-AT
    @REQUEST-LOCATION-CONSTRAINED
  @PROPOSE-PLAN
  [RUN-PLAN]
    @SEARCH-FOR-LOST-OBJECT
```

```
DRONE-D [SUB]                          Thoughts
I interpreted the input "I think I left my keys at home. Can you look around for them?" as @REQUEST-ACTION.
DANNY wants us to @SEARCH-FOR-LOST-OBJECT.
I'm going to wait for a plan from my team leader.
I interpreted the input "They are on a red keychain with a small flashlight." as @KEY.
I interpreted the input "I used them last night to open the front door, but they could be anywhere." as @UNLOCK-EVENT.
I interpreted the input "Let's search the apartment." as @PROPOSE-PLAN.
```

Fig. 5. A LEIA simulation system supplemented with under-the-hood panels that show traces of system functioning. The left-hand side of the interface shows the environment and the robots' actions in it; the middle panel shows the chat window; and the right-hand side shows four of the many under-the-hood panels that can be viewed during system operation.

At the point of the simulation captured by this screen shot, the ground vehicle has just proposed to the drone that they search the apartment. The TMRs (Recent) panel shows the ground vehicle's interpretation of Danny's most recent utterance. The upper Thoughts panel shows a trace of the ground vehicle's thoughts over the course of the multi-turn interaction, translated into English for the benefit of people viewing the demonstration (agents think using structures of the ontological metalanguage). The Agenda panel shows the agent's plan, with gray actions already having been completed and black ones in process. And the lower Thoughts panel shows a trace of the drone's thoughts throughout the interaction. Although we currently allow for only four panels to be shown at a time (a matter of screen real estate) others are available as well. For example, at a particular moment in time, the ground vehicle and drone are looking at a couch and a cabinet, respectively, as shown in Fig. 6. Note that vision meaning representations (VMRs) are structurally identical to text meaning representations, which in turn mirror the structures in the ontology.

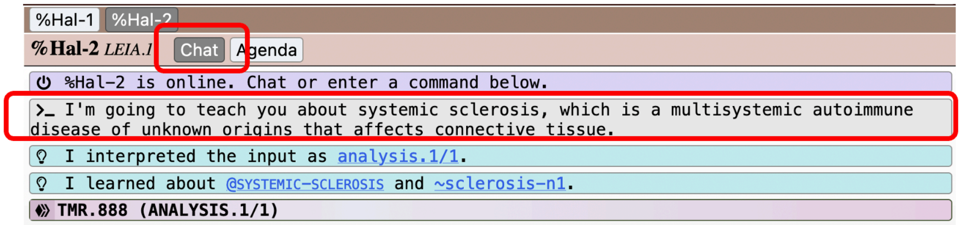


```
UGV-U [LEADER]                         VMRs (Recent)
COUCH.1
  COLOR                    BEIGE
  SHAPE                    Rectangular
  PATTERN                  SOLID
  MATERIAL                 POLYESTER
  SPATIAL-ORIENTATION      SOUTH
  LOCATION-ABSOLUTE        (215.38, 0.00, 51.00)
  ROTATION-ABSOLUTE        (0.00, 90.00, 0.00)
  COREFER                  #COUCH.1
LEIA.1
  COREFER                  #LEIA.1
  LOCATION                 (182.38, 3.30,
                           311.27)
  ORIENTATION              (0.00, 83.91, 0.00)
VISUAL-EVENT.1
  AGENT                    #LEIA.1
  THEME                    COUCH.1
```

```
DRONE-D [SUB]                          VMRs (Recent)
CABINETS.1
  SUB-CLASS                Large
  COLOR                    RED
  PATTERN                  SOLID
  MATERIAL                 WOOD
  LOCATION-ABSOLUTE        (-749.28, 16.94,
                           -740.44)
  ROTATION-ABSOLUTE        (0.00, 0.00, 0.00)
  COREFER                  #CABINETS.1
LEIA.1
  COREFER                  #LEIA.1
  LOCATION                 (-334.01, 450.00,
                           -624.74)
  ORIENTATION              (0.00, 0.00, 0.00)
VISUAL-EVENT.1
  AGENT                    #LEIA.1
  THEME                    CABINETS.1
```

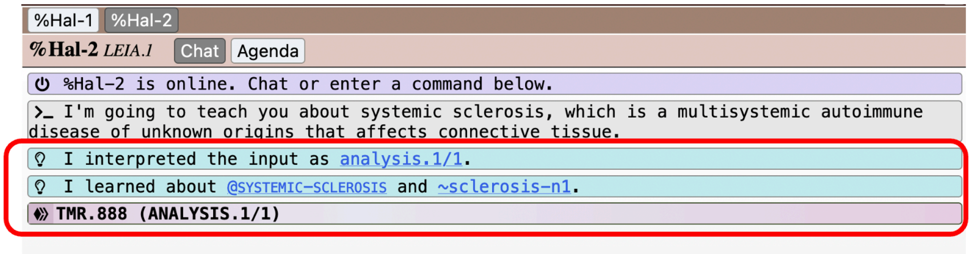
Fig. 6. Traces of vision meaning representations (VMRs).

Agent cognition can also be demonstrated from inside our development environment, called DEKADE. We will walk through a short demo of agent learning, using inline screen shots to avoid breaking the flow. Information that the agent (Hal-2) needs to learn is typed into the chat window.



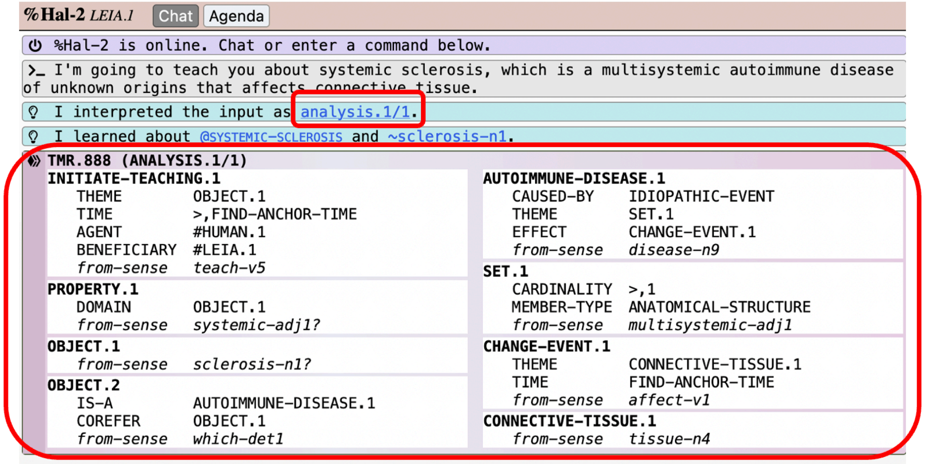
```
⏻ %Hal-2 is online. Chat or enter a command below.
>_ I'm going to teach you about systemic sclerosis, which is a multisystemic autoimmune
disease of unknown origins that affects connective tissue.
💡 I interpreted the input as analysis.1/1.
💡 I learned about @SYSTEMIC-SCLEROSIS and ~sclerosis-n1.
TMR.888 (ANALYSIS.1/1)
```

The agent generates a TMR, saved as analysis.1/1, and remembers that it learned a new ontological concept (SYSTEMIC-SCLEROSIS) and a new lexical sense (sclerosis-n1, which covers the whole collocation "systemic sclerosis").



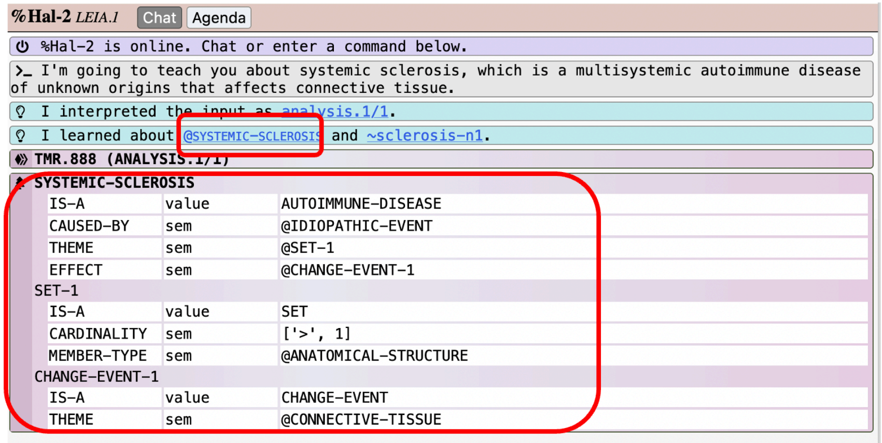
```
⏻ %Hal-2 is online. Chat or enter a command below.
>_ I'm going to teach you about systemic sclerosis, which is a multisystemic autoimmune
disease of unknown origins that affects connective tissue.
💡 I interpreted the input as analysis.1/1.
💡 I learned about @SYSTEMIC-SCLEROSIS and ~sclerosis-n1.
TMR.888 (ANALYSIS.1/1)
```

The language analysis can be viewed by clicking on analysis.1/1



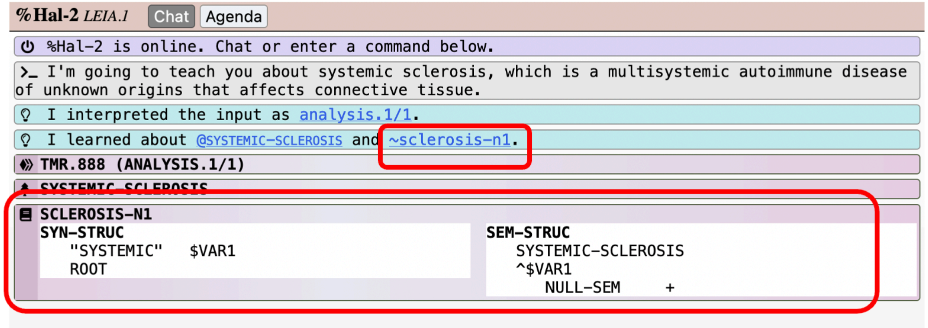
```
⏻ %Hal-2 is online. Chat or enter a command below.
>_ I'm going to teach you about systemic sclerosis, which is a multisystemic autoimmune disease
of unknown origins that affects connective tissue.
💡 I interpreted the input as analysis.1/1.
💡 I learned about @SYSTEMIC-SCLEROSIS and ~sclerosis-n1.
TMR.888 (ANALYSIS.1/1)
INITIATE-TEACHING.1                          AUTOIMMUNE-DISEASE.1
    THEME        OBJECT.1                        CAUSED-BY     IDIOPATHIC-EVENT
    TIME         >,FIND-ANCHOR-TIME              THEME         SET.1
    AGENT        #HUMAN.1                        EFFECT        CHANGE-EVENT.1
    BENEFICIARY  #LEIA.1                         from-sense    disease-n9
    from-sense   teach-v5                    SET.1
PROPERTY.1                                       CARDINALITY   >,1
    DOMAIN       OBJECT.1                        MEMBER-TYPE   ANATOMICAL-STRUCTURE
    from-sense   systemic-adj1?                  from-sense    multisystemic-adj1
OBJECT.1                                     CHANGE-EVENT.1
    from-sense   sclerosis-n1?                   THEME         CONNECTIVE-TISSUE.1
OBJECT.2                                         TIME          FIND-ANCHOR-TIME
    IS-A         AUTOIMMUNE-DISEASE.1            from-sense    affect-v1
    COREFER      OBJECT.1                    CONNECTIVE-TISSUE.1
    from-sense   which-det1                      from-sense    tissue-n4
```

The learned ontological frame can be viewed by clicking on @SYSTEMIC-SCLEROSIS.



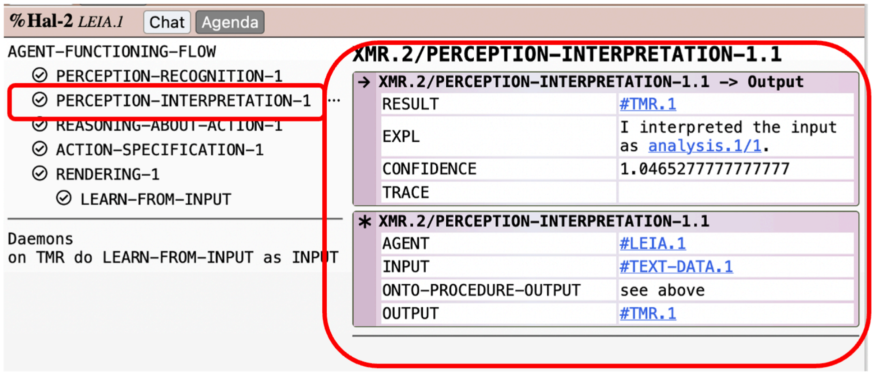
```
%Hal-2 is online. Chat or enter a command below.
>_ I'm going to teach you about systemic sclerosis, which is a multisystemic autoimmune disease
of unknown origins that affects connective tissue.
I interpreted the input as analysis.1/1.
I learned about @SYSTEMIC-SCLEROSIS and ~sclerosis-n1.
TMR.888 (ANALYSIS.1/1)
SYSTEMIC-SCLEROSIS
  IS-A          value     AUTOIMMUNE-DISEASE
  CAUSED-BY     sem       @IDIOPATHIC-EVENT
  THEME         sem       @SET-1
  EFFECT        sem       @CHANGE-EVENT-1
SET-1
  IS-A          value     SET
  CARDINALITY   sem       ['>', 1]
  MEMBER-TYPE   sem       @ANATOMICAL-STRUCTURE
CHANGE-EVENT-1
  IS-A          value     CHANGE-EVENT
  THEME         sem       @CONNECTIVE-TISSUE
```

And the learned lexical sense can be viewed by clicking on ~sclerosis-n1.



```
%Hal-2 is online. Chat or enter a command below.
>_ I'm going to teach you about systemic sclerosis, which is a multisystemic autoimmune disease
of unknown origins that affects connective tissue.
I interpreted the input as analysis.1/1.
I learned about @SYSTEMIC-SCLEROSIS and ~sclerosis-n1.
TMR.888 (ANALYSIS.1/1)
SYSTEMIC-SCLEROSIS
SCLEROSIS-N1
SYN-STRUC                               SEM-STRUC
  "SYSTEMIC"   $VAR1                      SYSTEMIC-SCLEROSIS
  ROOT                                    ^$VAR1
                                            NULL-SEM     +
```

The agent's agenda during learning can also be viewed, and each of the remembered events opens up into a detailed trace of the associated processing.



```
AGENT-FUNCTIONING-FLOW                 XMR.2/PERCEPTION-INTERPRETATION-1.1
  PERCEPTION-RECOGNITION-1             XMR.2/PERCEPTION-INTERPRETATION-1.1 -> Output
  PERCEPTION-INTERPRETATION-1 ...        RESULT                 #TMR.1
  REASONING-ABOUT-ACTION-1               EXPL                   I interpreted the input
  ACTION-SPECIFICATION-1                                        as analysis.1/1.
  RENDERING-1                            CONFIDENCE             1.0465277777777777
    LEARN-FROM-INPUT                     TRACE
                                       XMR.2/PERCEPTION-INTERPRETATION-1.1
Daemons                                  AGENT                  #LEIA.1
on TMR do LEARN-FROM-INPUT as INPUT      INPUT                  #TEXT-DATA.1
                                         ONTO-PROCEDURE-OUTPUT  see above
                                         OUTPUT                 #TMR.1
```

When more information is provided about the newly-learned concept, its ontological description becomes correspondingly more detailed.

Static screen shots with short descriptions don't do justice to how effective demonstration systems are when supplemented by under-the-hood panels. Of course, there is no end to how user-friendly they could be made given additional resources.

5. Final Thoughts

Although we have not emphasized the *shapes* analogy in this paper, we will use it to wrap up the discussion. The *shapes* orientation says: Let's exploit the fact that human knowledge and reasoning have long been conceptualized as highly structured. Let's focus on typical cases first, since that will go a long way to making agents useful. Let's enable agents to recover from atypical cases using generalized recovery strategies (getting by with incomplete understanding; learning something new; asking a human for help). Let's think about models as pictures, be they diagrams or templates, so we can clearly understand, remember, and explain them to all relevant stakeholders. And let's open the hood on agent operation to prove that our demonstration systems actually implement our theoretical claims and have the potential to rise to the challenge of scalability and real-world utility.

Acknowledgements

This research was supported in part by Grants N00014-16-1-2118 and N00014-17-1-2218 from the U.S. Office of Naval Research. Any opinions or findings expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

Chomsky, N. (1957). *Syntactic Structures*. Mouton.

Connell, L., & Lynott, D. (2024). What can language models tell us about Human Cognition? *Current Directions in Psychological Science*, *33*(3), 181-189.

Cuskley, C., Woods, R., & Flaherty, M. (2024). The limitations of large language models for understanding human language and cognition. *Open Mind* 8: 1058-1083.

Fillmore, C. J., & Baker, C. F. (2009). A frames approach to semantic analysis. In B. Heine & H. Narrog (Eds.), *The Oxford Handbook of Linguistic Analysis* (pp. 313–340). Oxford University Press.

Gentner, D. & Smith, L. A. (2013). Analogical learning and reasoning. In D. Reisberg (Ed.), *The Oxford Handbook of Cognitive Psychology*, 668–681. New York, NY: Oxford University Press.

Hoffmann, T., & Trousdale, G. (Eds.). (2013). *The Oxford Handbook of Construction Grammar*. Oxford University Press.

Kapoor, S., et al. (2024). Large language models must be taught to know what they don't know. *Advances in Neural Information Processing Systems* 37: 85932-85972.

Marcus, G. (2025). Game over for pure LLMs. Even Turing Award winner Rich Sutton has gotten off the bus. Marcus on AI (in Substack). September 26.

McShane, M., & Nirenburg, S. (2021). *Linguistics for the Age of AI*. The MIT Press. Available open-access at https://direct.mit.edu/books/oa-monograph/5042/Linguistics-for-the-Age-of-AI.

McShane, M., Nirenburg, S., & English, J. (2024). *Agents in the Long Game of AI: Computational cognitive modeling for trustworthy, hybrid AI*. MIT Press. Available open-access at https://direct.mit.edu/books/oa-monograph/5833/Agents-in-the-Long-Game-of-AIComputational.

McShane, M., Nirenburg, S., English, J., & Oruganti, S. (2025). Pursuing actionable perception interpretation in cognitive robotic systems. *Advances in Cognitive Systems* 2025.

McShane, M.. Nirenburg, S., Goodman, K., Oruganti, S., and English, J. (Forthcoming). Trust through explainability in cognitive agents, in *EXPLAINS 2024*. Springer.

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Prentice-Hall.

Nirenburg, S., & Raskin, V. (2004). *Ontological Semantics*. MIT Press.

Nirenburg, S., McShane, M., & English, J. (2023). Content-centric computational cognitive modeling. *Advances in Cognitive Systems*, vol. 10. http://www.cogsys.org/journal/volume10/article-10-6.pdf

Oruganti, S., Nirenburg, S., McShane, M., English, J., Roberts, M., & Arndt, C. (2024a). HARMONIC: A framework for explanatory cognitive robots. *Proceedings of ICRA@40*. Rotterdam, The Netherlands.

Oruganti, S., Nirenburg, S., McShane, M., English, J., Roberts, M., & Arndt, C. (2024b). HARMONIC: Cognitive and Control Collaboration in Human-Robotic Teams. arXiv preprint arXiv:2409.18047.

Rosch, E. H. (1973). Natural categories. *Cognitive Psychology*, 4(3): 328–*350.* doi*:*10.1016/0010-0285(73)90017-0. ISSN 0010-0285.

Schank, R. (1982). *Dynamic Memory: A theory of learning in computers and people.* Cambridge University Press.

Schank, R., & Abelson, R. P. (1977). *Scripts, Plans, Goals and Understanding: An inquiry into human knowledge structures*. Erlbaum.

Sun, R., Slusarz, P., & Terry, C. (2005). The interaction of the explicit and the implicit in skill learning: A dual-process approach. *Psychological Review*, 112(1), 159–192.

Sung, J.Y., Harris, O.K., Hensley, N.M., Chemero, A.P., Morehouse, N.I. (2021). Beyond cognitive templates: Re-examining template metaphors used for animal recognition and navigation. *Integrative and Comparative Biology*, 61(3): 825-841.

Thalmann, M., Souza, A.S., Oberauer, K. (2019). How does chunking help working memory? *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 45(1): 37-55.

Xu, Ziwei, Jain, S., & Kankanhalli, M. (2024). Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.