Cognitive Architectures for Robust and Reliable Robotics

Esther Aguado
July 2025

Intelligent Robotics Lab
Universidad Rey Juan Carlos

2^{nd} ACM SIGSOFT Summer School for Software Engineering in Robotics

TU Delft
CORESENSE

About me

- Robotics teacher & researcher at URJC
- 5+ years in intelligent, robust robotic systems
- PhD in Automatic Control and Robotics from UPM
- Visiting researcher at TU Delft Cognitive Robotics Lab
- Focus: self-awareness, planning, deliberation

My journey into the topic

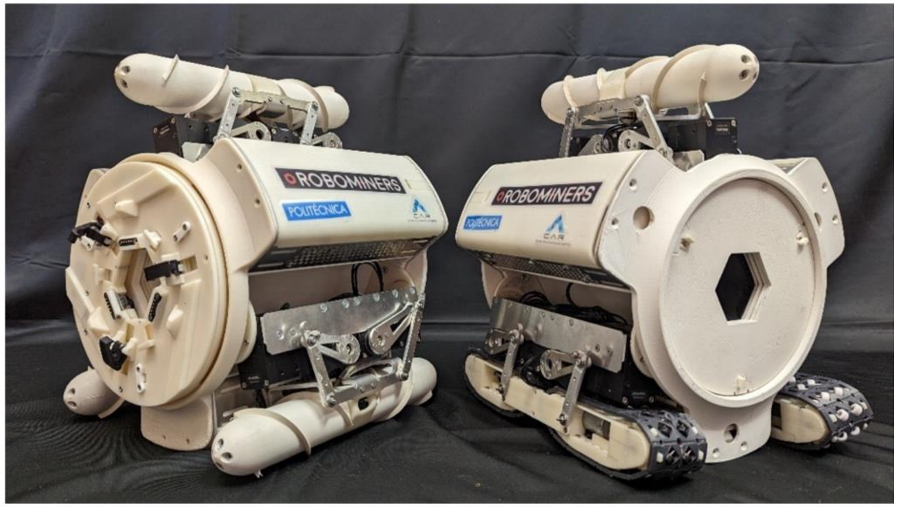
CORESENSE

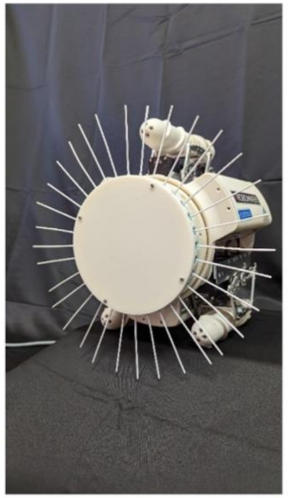
My journey into the topic

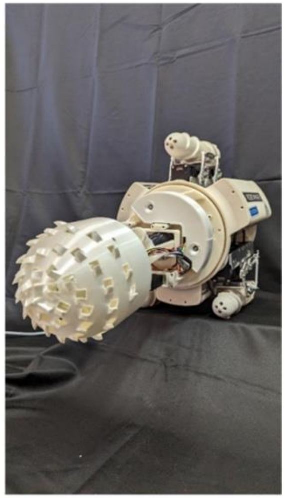
ROBO MINERS

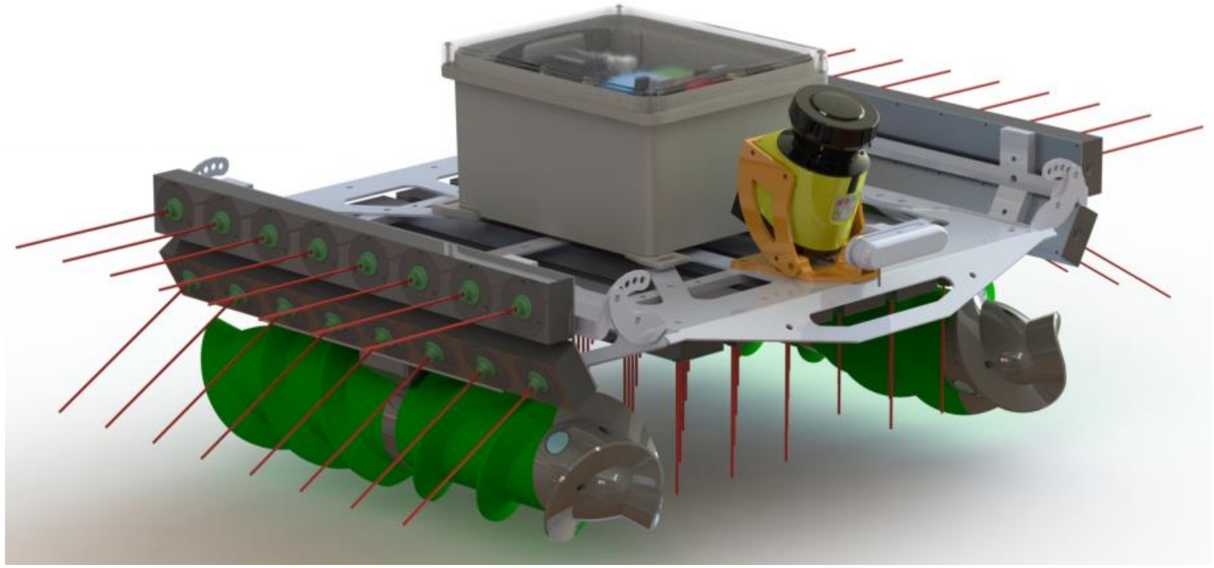
Intelligent Robotics Lab

My journey into the topic









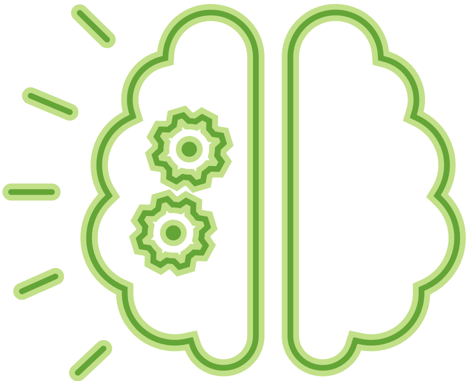
Wishlist

- Model the robot, its mission, and its environment
- Enable **adaptive behaviour** in challenging conditions
- Support different systems
- Promote **understanding**, not just action
- Robust **autonomy**

CORESENSE

Wishlist

- Model the robot, its mission, and its environment
- Enable **adaptive behaviour** in challenging conditions
- Support different systems
- Promote **understanding**, not just action
- Robust **autonomy**



Cognitive Architecture

Intelligent Robotics Lab

Outline

Part 1: Foundations of Cognitive Architectures

- What are Cognitive Architectures?
- Core capabilities
- Classical examples: SOAR, ACT-R, LIDA

Part 2: Deliberation in Robotics

- CRAM / KnowRob
- SkiROS
- SysSelf

Part 3: What is Next

- CoreSense
- Limitations and Future work
- Conclusions

Part 1.1:

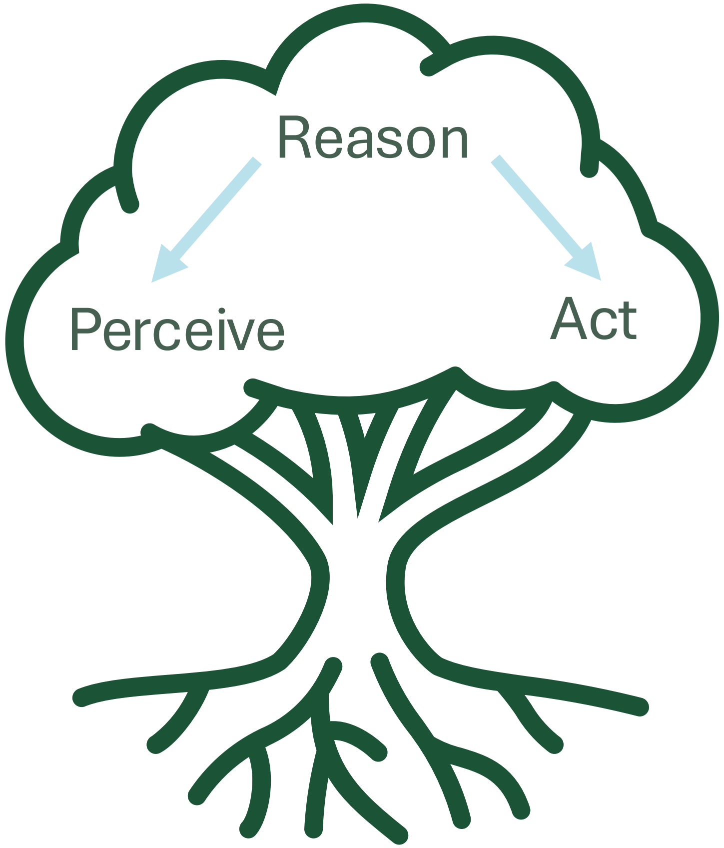
Foundations of Cognitive Architectures

What are Cognitive Architectures?

What is a cognitive architecture

Reusable blueprint that defines the core components of an intelligent system

Stable over time



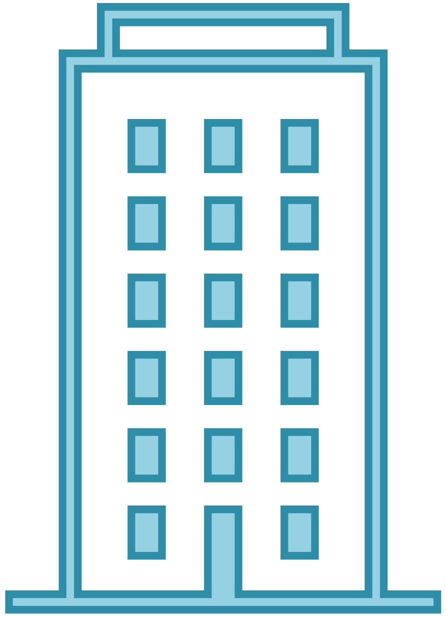
Applicable to different tasks and/or domains

What is a cognitive architecture

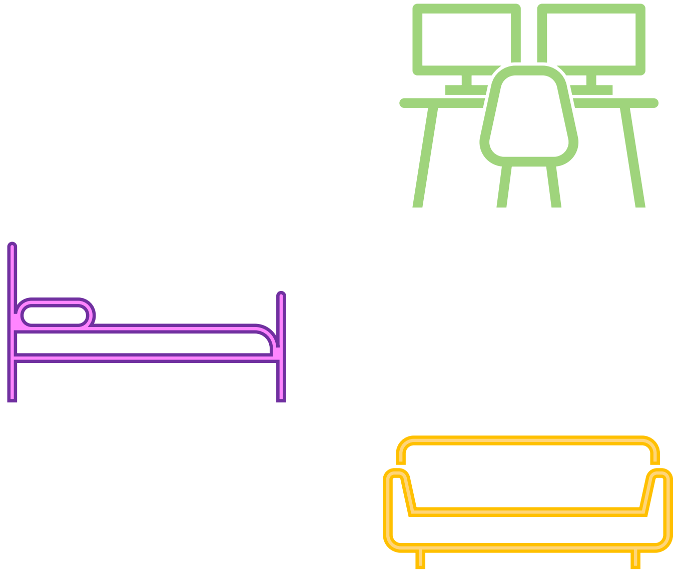
Supported knowledge:

- **Memory** (short- and long-term): Storage of beliefs, goals, and knowledge
- **Representation**: Internal models of the environment, self, or task
- **Functional Processes**: Mechanisms that operate over representations (e.g., reasoning, planning, learning)

The building analogy



Cognitive architecture



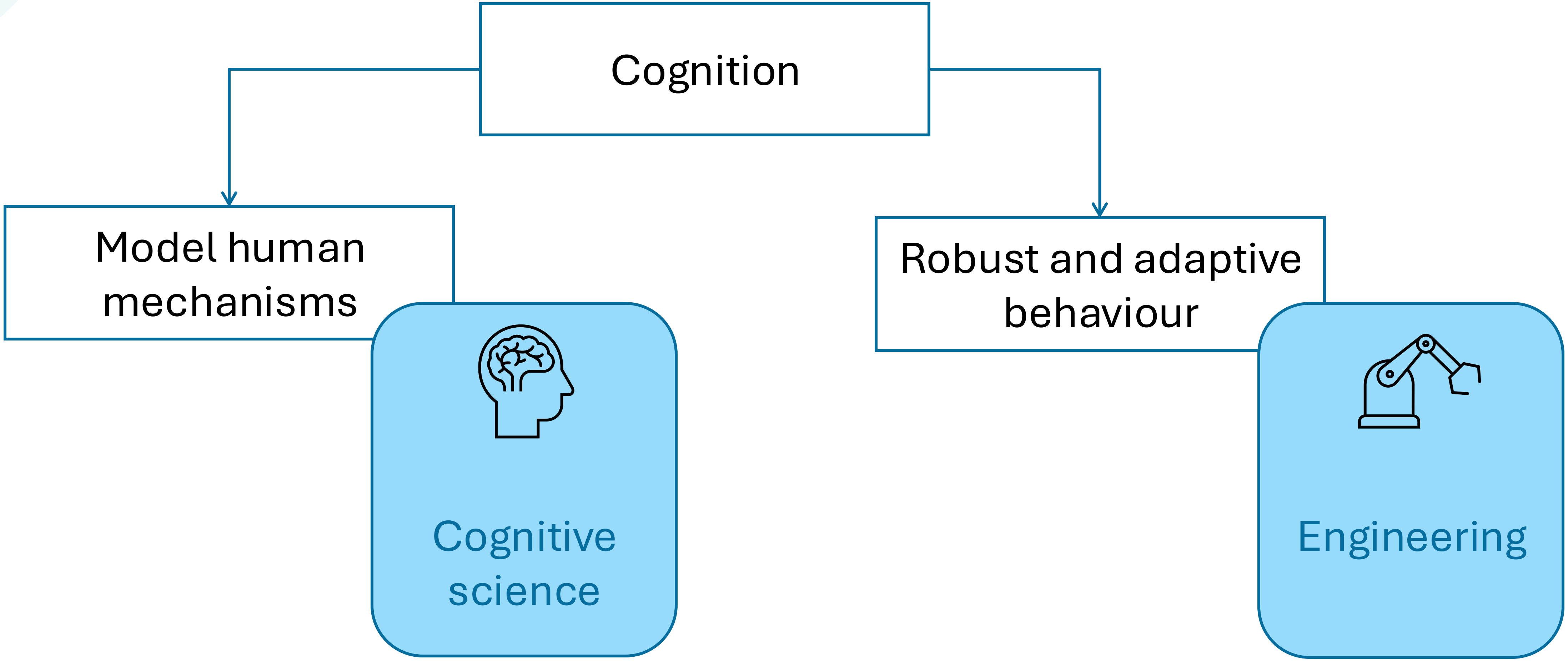
Domain specific behaviors, skills, algorithms

CORESENSE

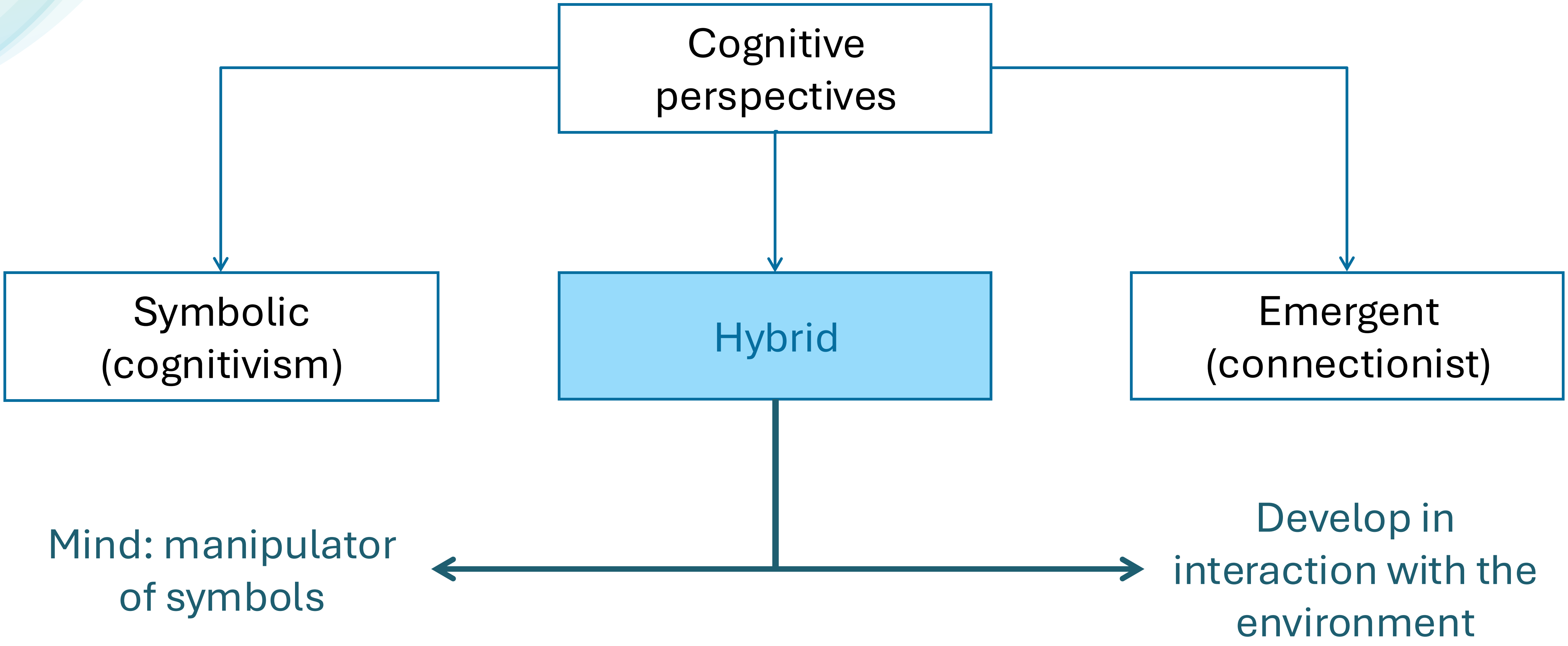
Intelligent Robotics Lab

The building analogy

Approaches in cognitive architectures



Perspectives in cognitive architectures



CORESENSE

Intelligent vs cognitive system

What's the difference?

- Both may use memory, control, I/O, internal models
- But cognitive systems evolve over time
- They update internal knowledge and adapt behaviours
- Intelligent systems are often fixed and task-specific

Intelligent Robotics Lab

Intelligent vs cognitive system

- Cognitive systems are not just pipelines → integrated systems
- They must manage and use different types of knowledge:
  - Perception: external world
  - Planning: possible futures
  - Memory/Learning: past experiences
  - Communication: coordination

Cognitive system core: Knowledge

- How does the system access knowledge?
- How does it reason about it?
- How does it use it to make informed decisions?

A cognitive system must know ***when*** *and* ***how*** *to use* ***what*** *type of knowledge, depending on the task and context.*

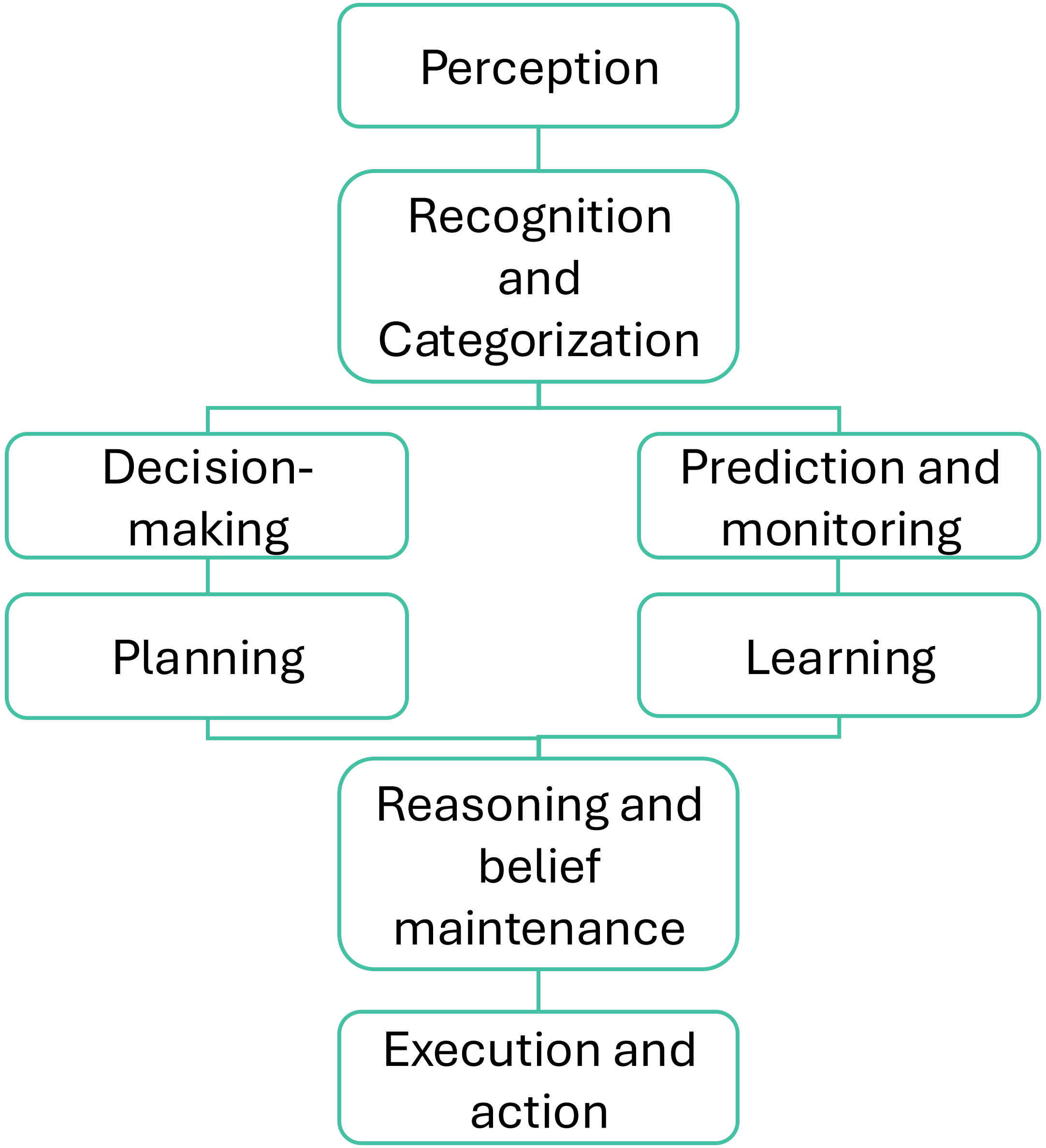
Part 1.2:

Foundations of Cognitive Architectures

Core Capabilities

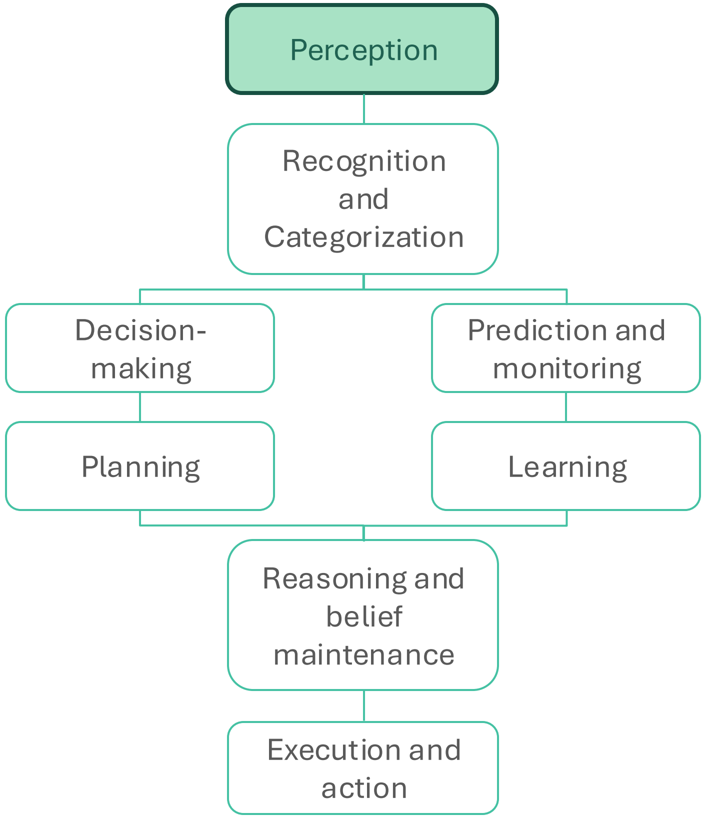
Core capabilities

Langley et al., Cognitive architectures: Research issues and challenges (2006)



Perception : Transforming sensory data

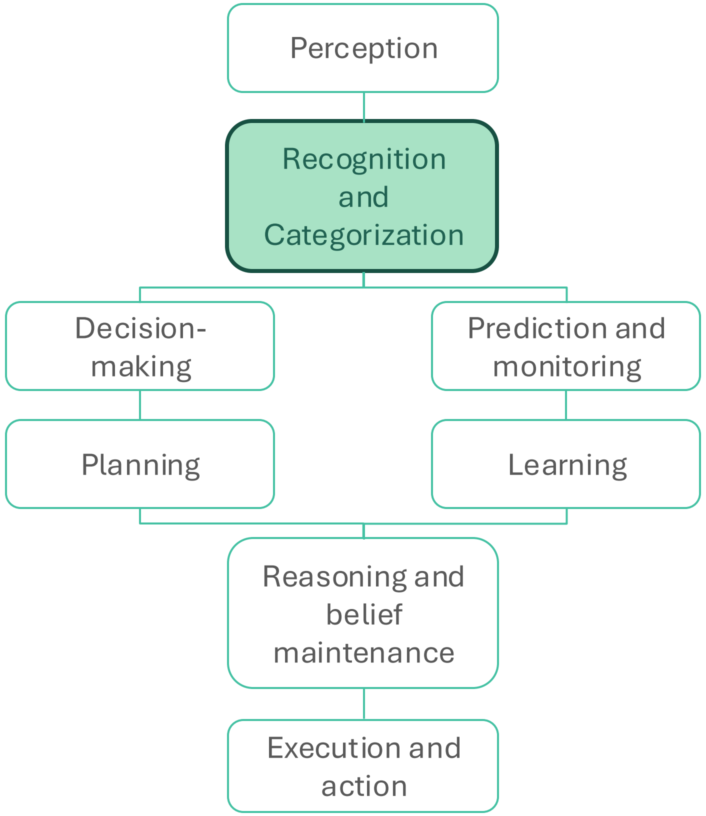
- Beyond raw data: **Convert** sensor inputs into usable representations
- Attention management: Allocate limited perceptual resources to detect and **prioritize** relevant signals
- Signal vs. noise: Identify **critical information** in complex, cluttered environments
- Understanding: **interpreting** what's perceived to support reasoning and action



Recognition and Categorization: From data to concepts

Abstract processed perceptions:

- Integrate multi-sensor data in a unified model
- Pattern matching
- Examples:
  - Reading: letters → words → meaning
  - Service robot: kitchen area vs. seating area → correct delivery



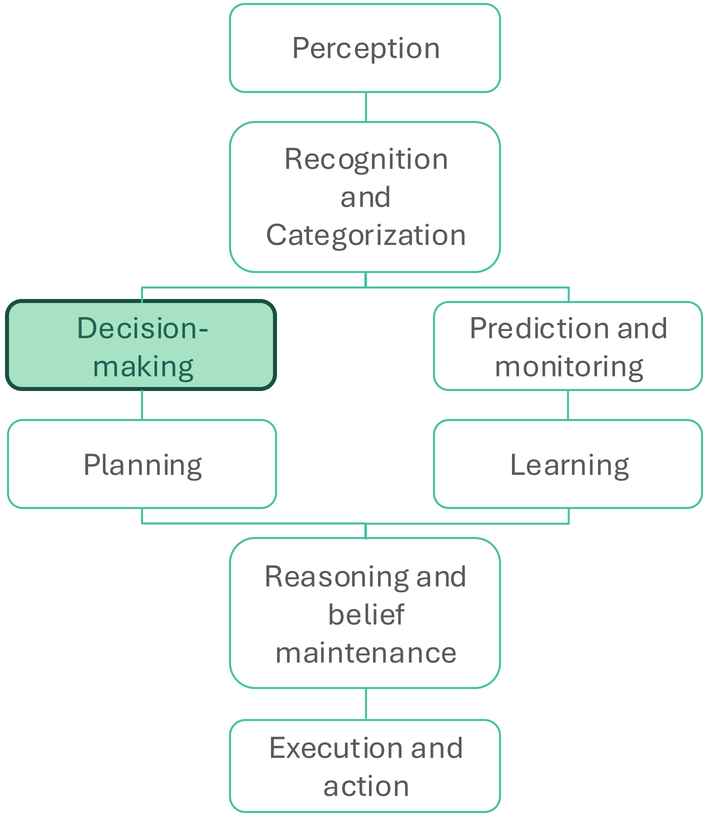
Decision-making: Reactive vs. Deliberative

Reactive decisions:

- Fast, context-driven
- Based on recognize-act cycles

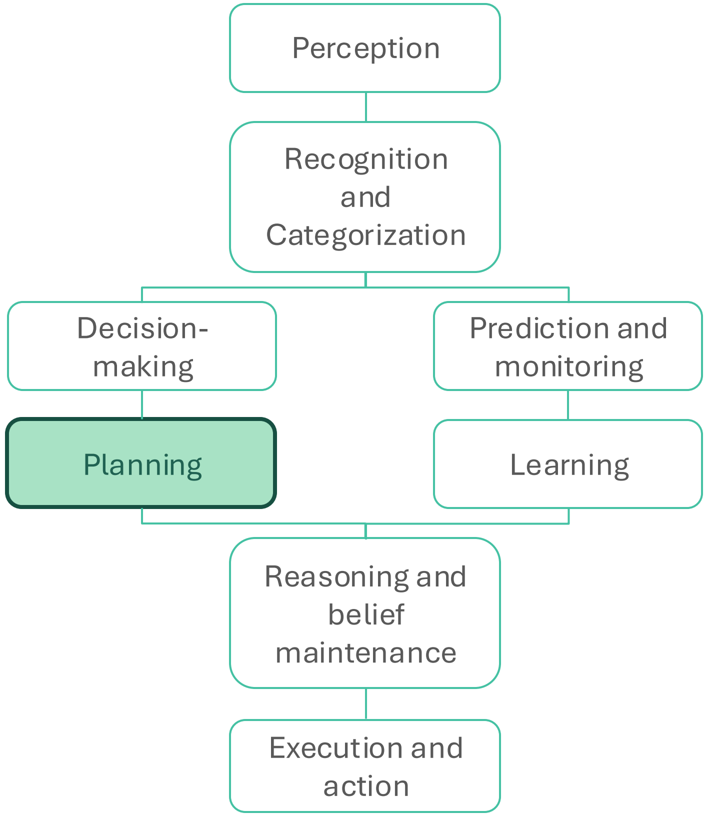
Deliberative decisions:

- Slow, goal-oriented reasoning
- Evaluate possible actions against goals and constraints



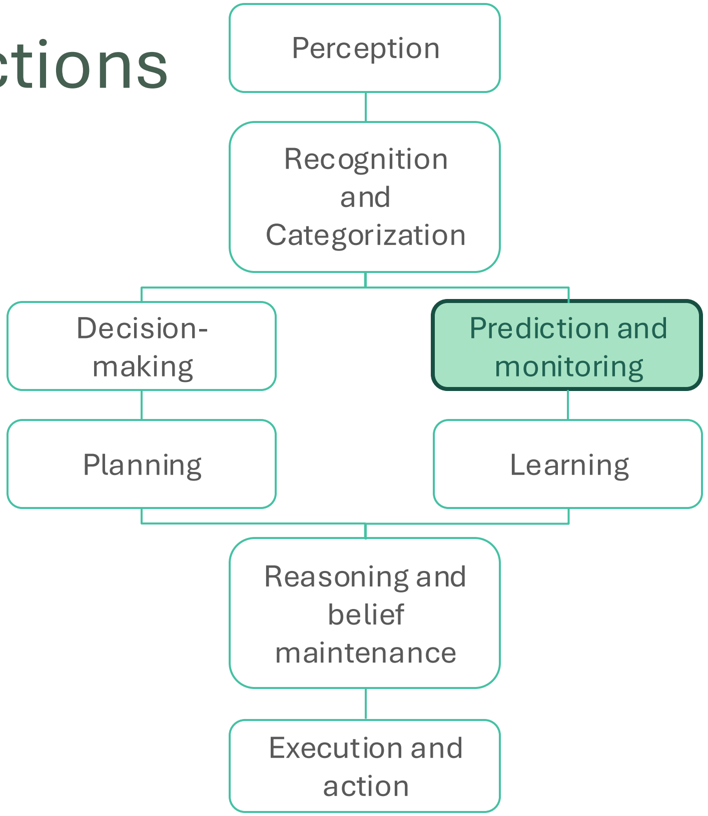
Planning: Goal-directed strategies

- Achieve goals in new situations
- Model the world: predict action effects
- Plan representation: ordered actions + expected effects → support subsequent steps
- Plan execution: translate high-level steps into low-level motor commands
- Replanning: not just fault-tolerance, also better ways to reach goals



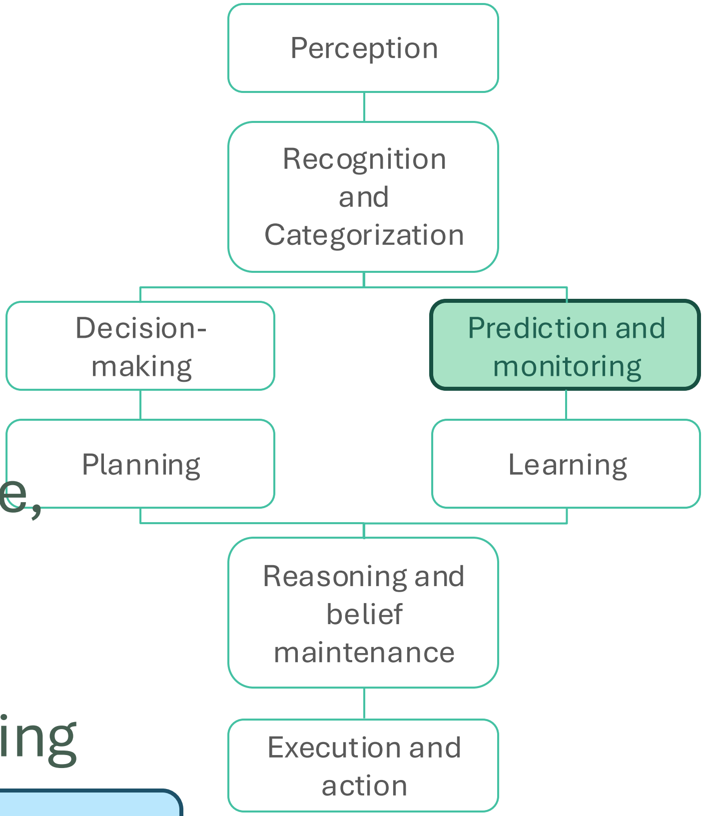
Predicting outcomes & monitoring execution

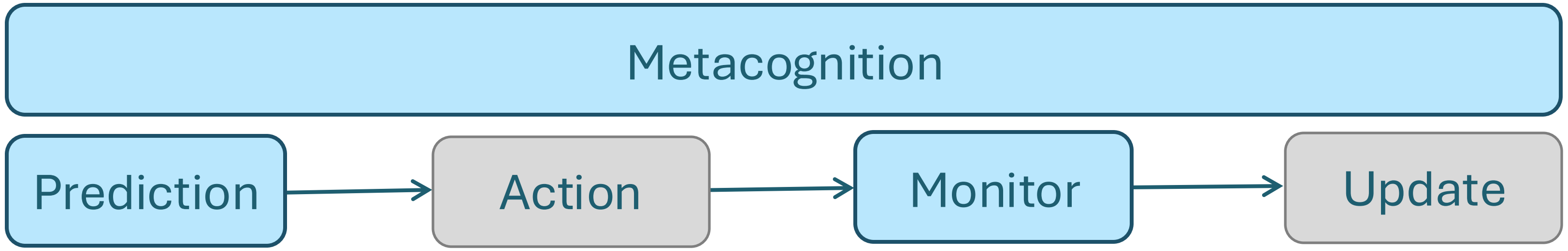
- Prediction: use models to estimate effects of actions
  - Map (state × action) → expected outcome
  - Explicit action models (e.g., classical planners)
- Monitoring
  - Compare predicted vs. actual outcomes
  - Trigger adaptation or replanning if needed



Predicting Outcomes & Monitoring Execution

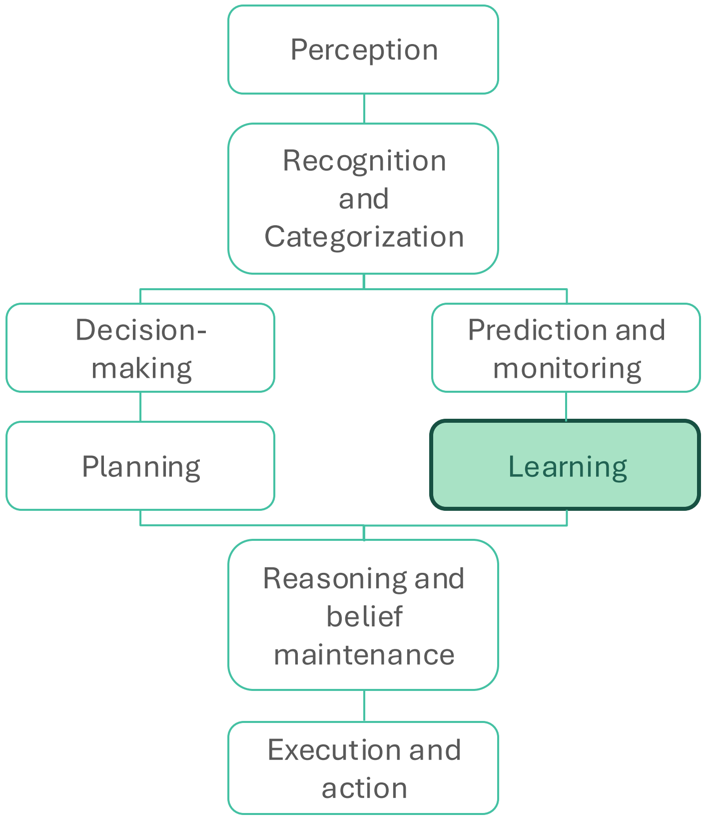
- Learning through monitoring
  - Update models when predictions fail
- Metacognition
  - Reflect on internal processes (resources, confidence, progress)
  - Enables self-awareness and adaptive decision-making





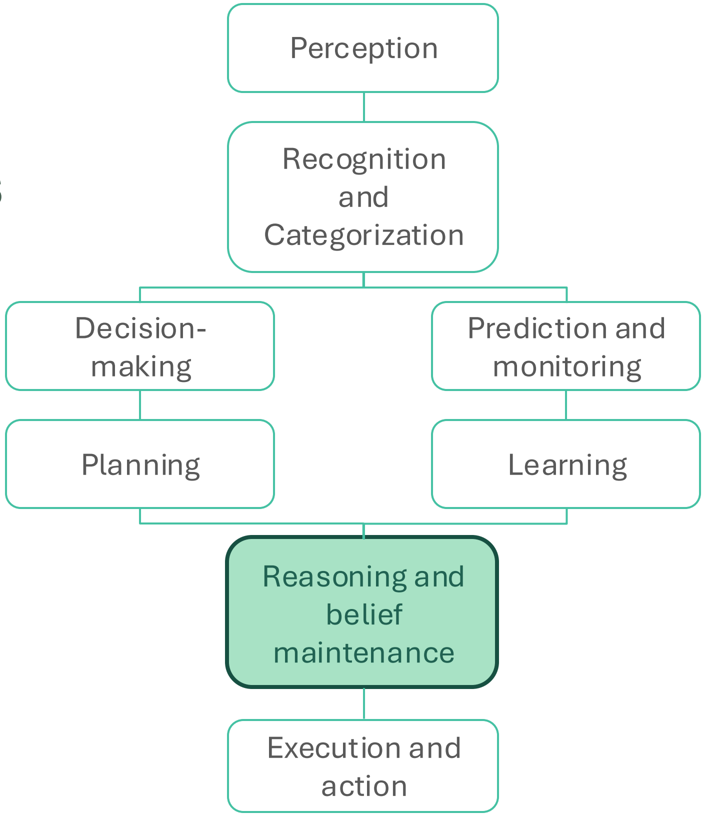
Learning

- Core process:
  - Remember: Store past experiences
  - Reflect: Analyse to find patterns
  - Generalize: Apply insights to new situations
- Learning strategies:
  - Specific experiences that may be generalized later
  - Learning from experience
  - Metareasoning for self-directed, strategic learning



Reasoning: Drawing conclusions from beliefs

- Reasoning vs. Planning
  - Planning: Select actions in the world to achieve goals
  - Reasoning: Derives internal conclusions from beliefs
- Knowledge representation: encode relationships
- Inference mechanisms:
  - Primarily deductive reasoning
  - May also support abductive or probabilistic inference

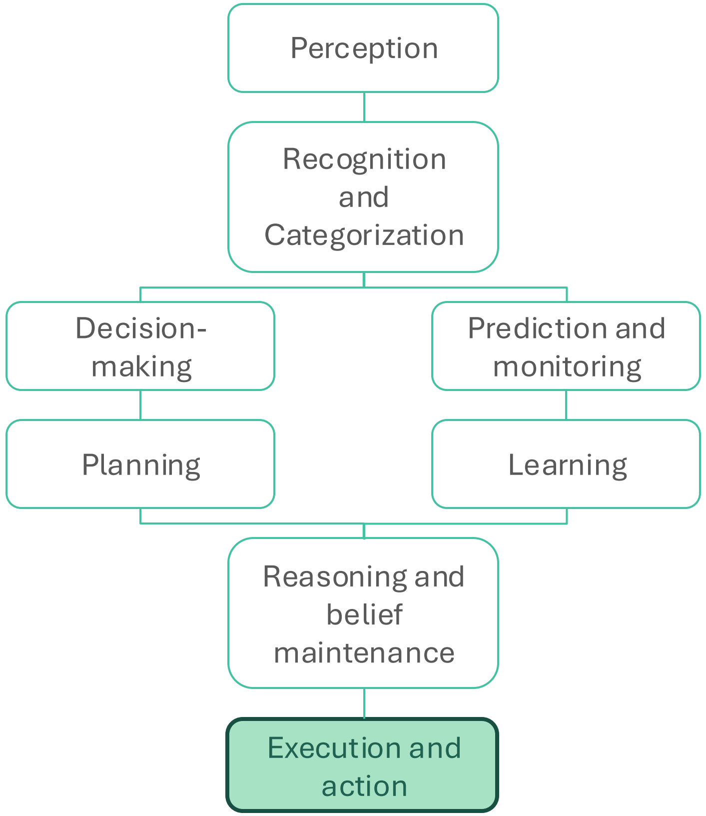


CORESENSE

Intelligent Robotics Lab

Execution: Turning decisions into actions

- Goal: Ensure decisions lead to desired real-world results → how to act
- Execution Modes:
  - Closed-loop (reactive): continuous feedback & adjustment
    - Uncertain or dynamic environments
  - Open-loop (automatized)
    - Stored procedure



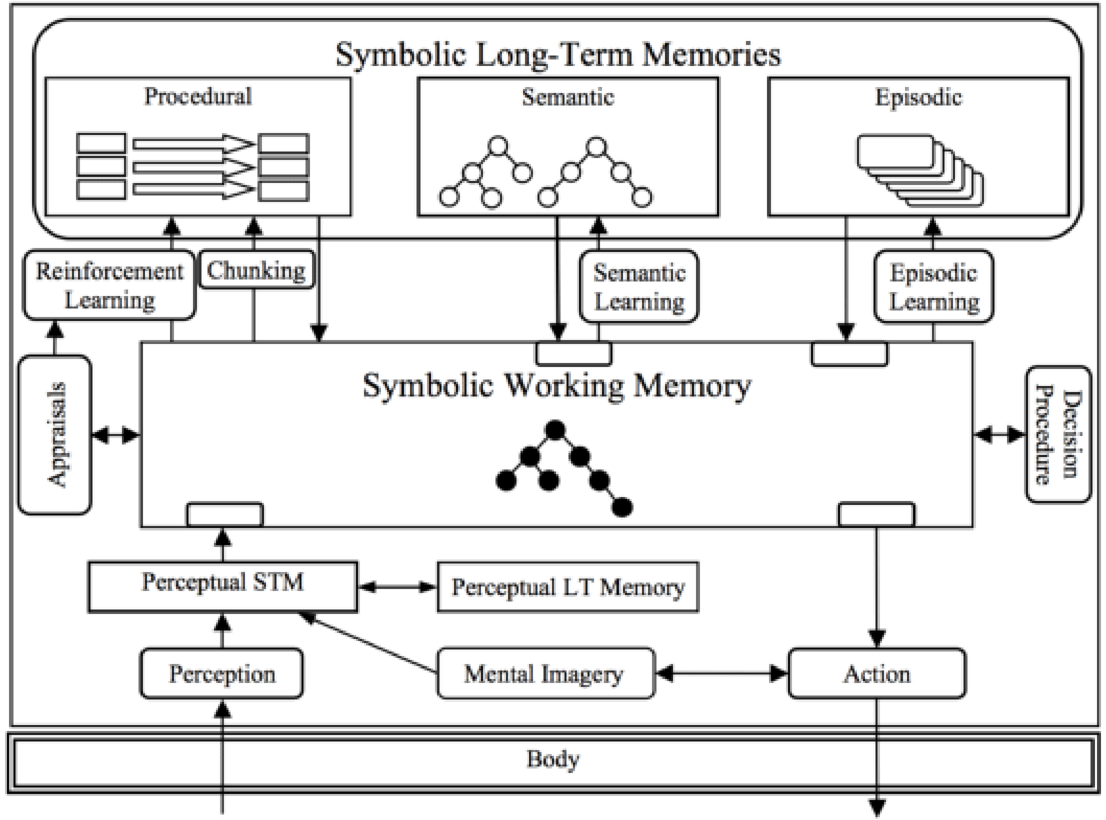
Part 1.3:

Foundations of Cognitive Architectures

Classical examples

SOAR

- Developed in the 1980s to model all aspects of cognition
- Key Features:
  - Symbolic knowledge representation
  - Problem solving via production rules
  - Learning through chunking (creating new rules from experience)



Introduction to the Soar Cognitive Architecture, Laird (2022)
https://arxiv.org/pdf/2205.03854

Memory systems in SOAR

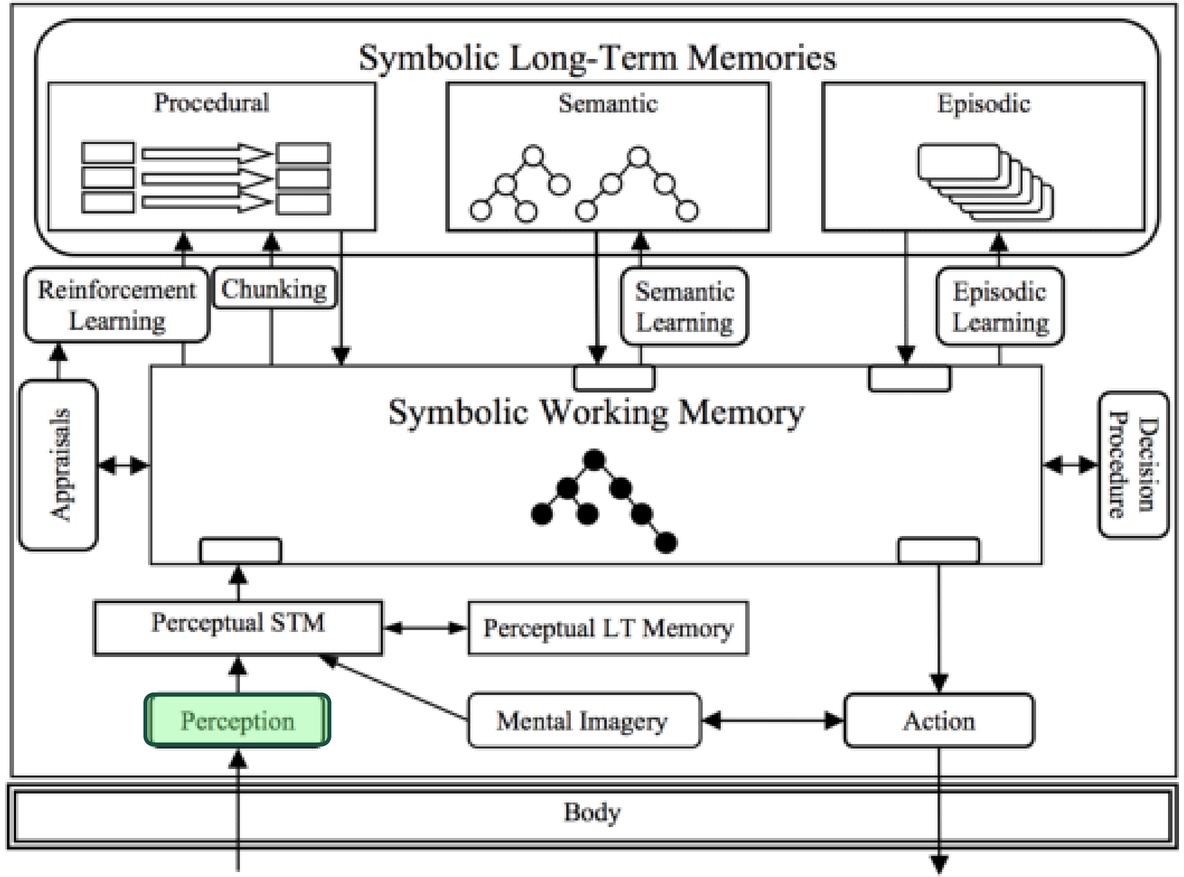
- Procedural Memory: rules and skills
- Semantic Memory: general knowledge
- Episodic Memory: past experiences
- Working Memory: active beliefs and goals

CORESENSE

SOAR: Perception and the Spatial Visual System

- Processes 2D and 3D visual input into symbolic form
- Mapped Capabilities:
  - Perception
  - Recognition and categorization

Image → Object o45 has color g35

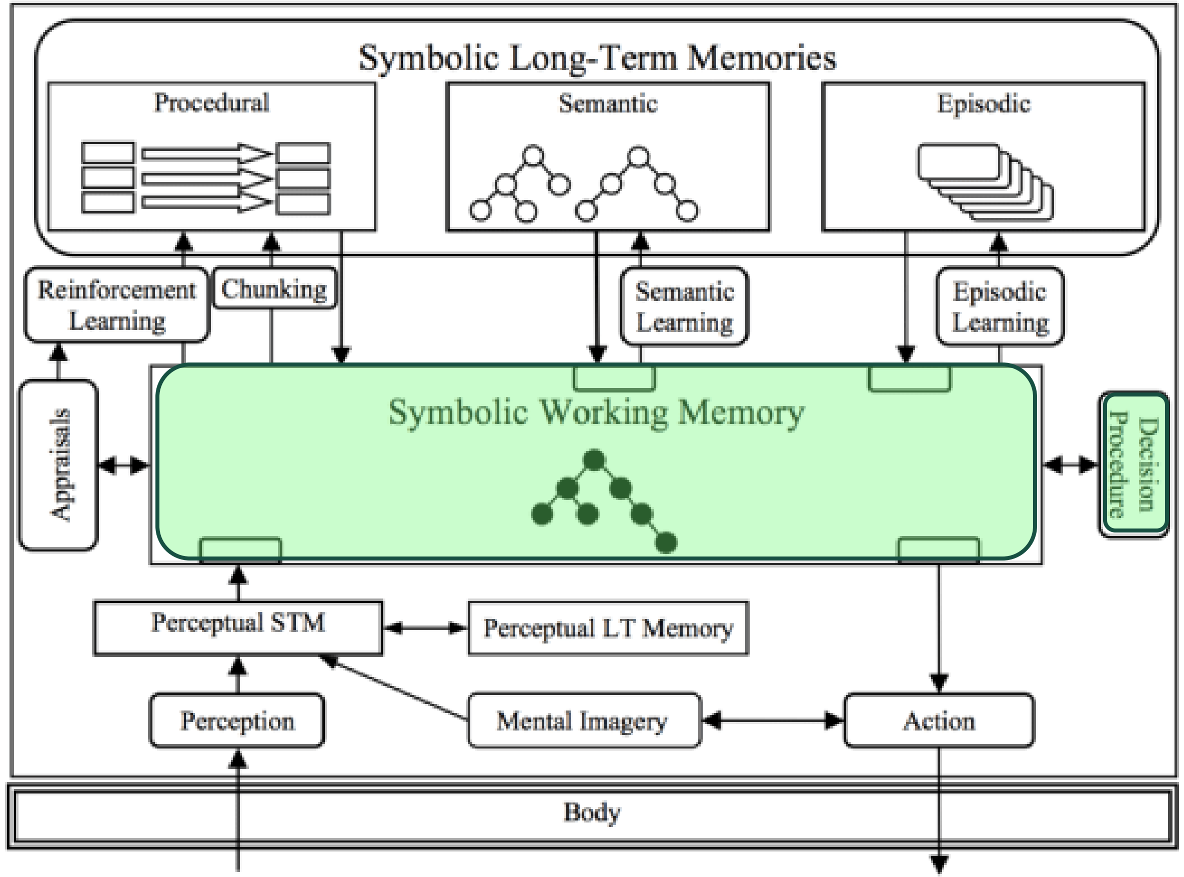


Introduction to the Soar Cognitive Architecture, Laird (2022)
https://arxiv.org/pdf/2205.03854

Intelligent Robotics Lab

SOAR: Reasoning and Decision-Making

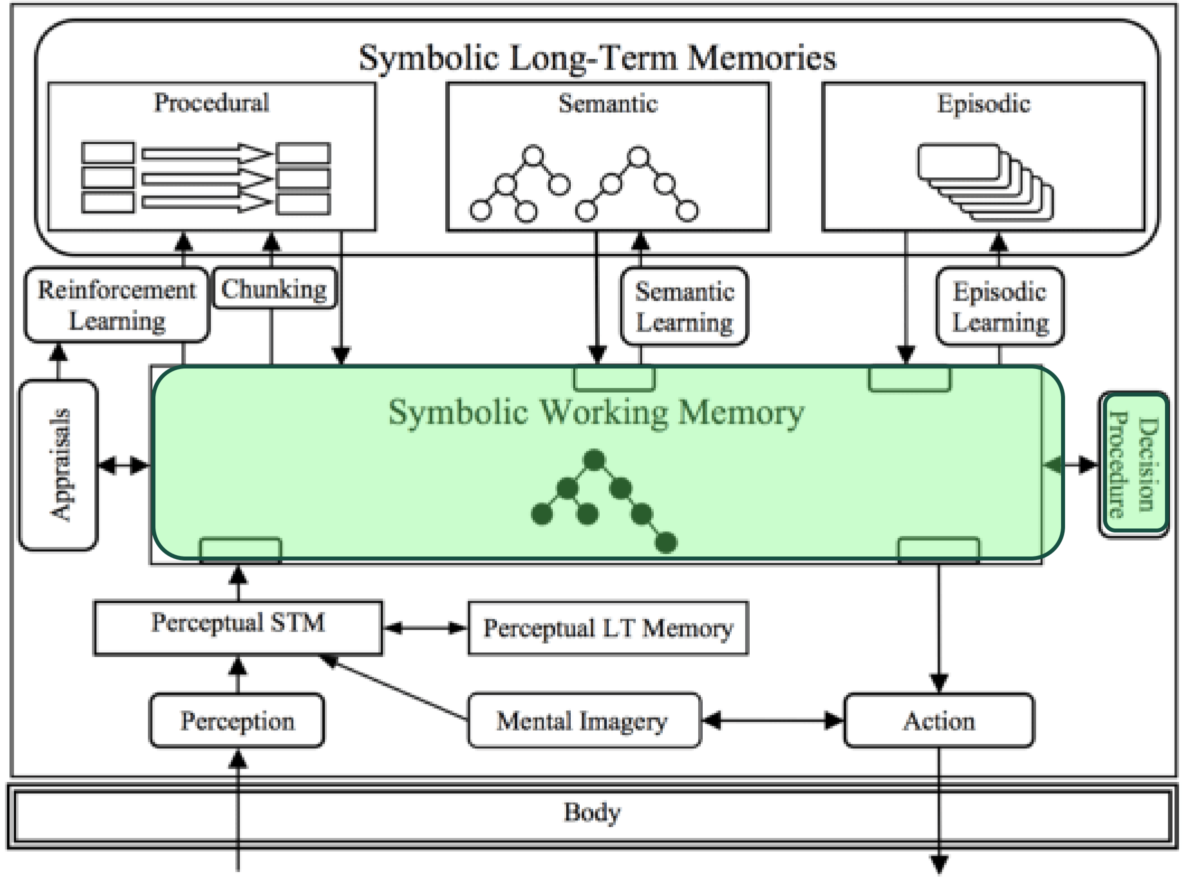
- Rules to elaborate the current state:
  - Adds beliefs
  - Evaluates conditions
  - Proposes operators (possible actions)
- If no clear choice → Impasse
  - Triggers a substate (a new reasoning context)
  - Allows deeper reflection on missing or conflicting knowledge



Introduction to the Soar Cognitive Architecture, Laird (2022)
https://arxiv.org/pdf/2205.03854

SOAR: Planning

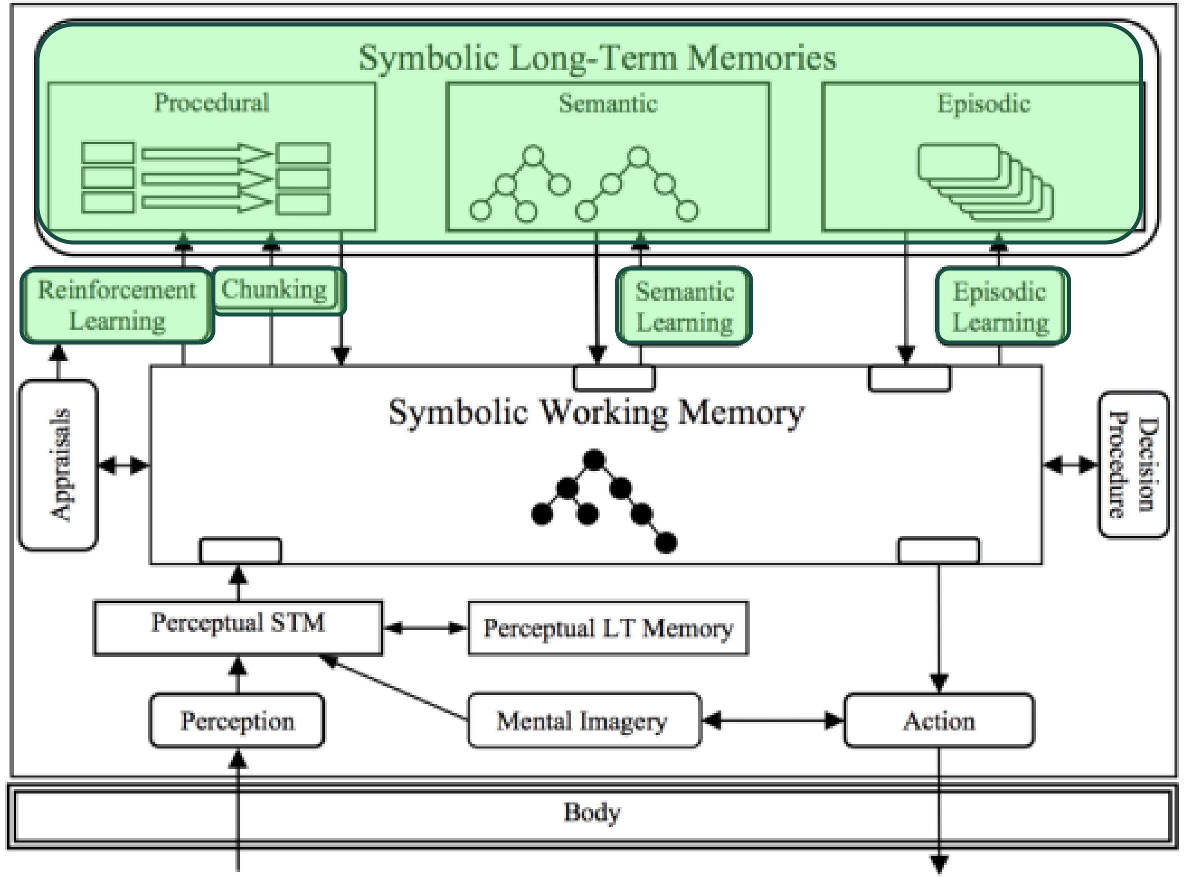
- Hierarchical and flexible planning
  - Decomposed goals
- Separate reasoning spaces
  - Each sub-state as a mental workspace
- Real-time adaptability
- Result: Plans are built dynamically, step by step



Introduction to the Soar Cognitive Architecture, Laird (2022)
https://arxiv.org/pdf/2205.03854

SOAR: Learning

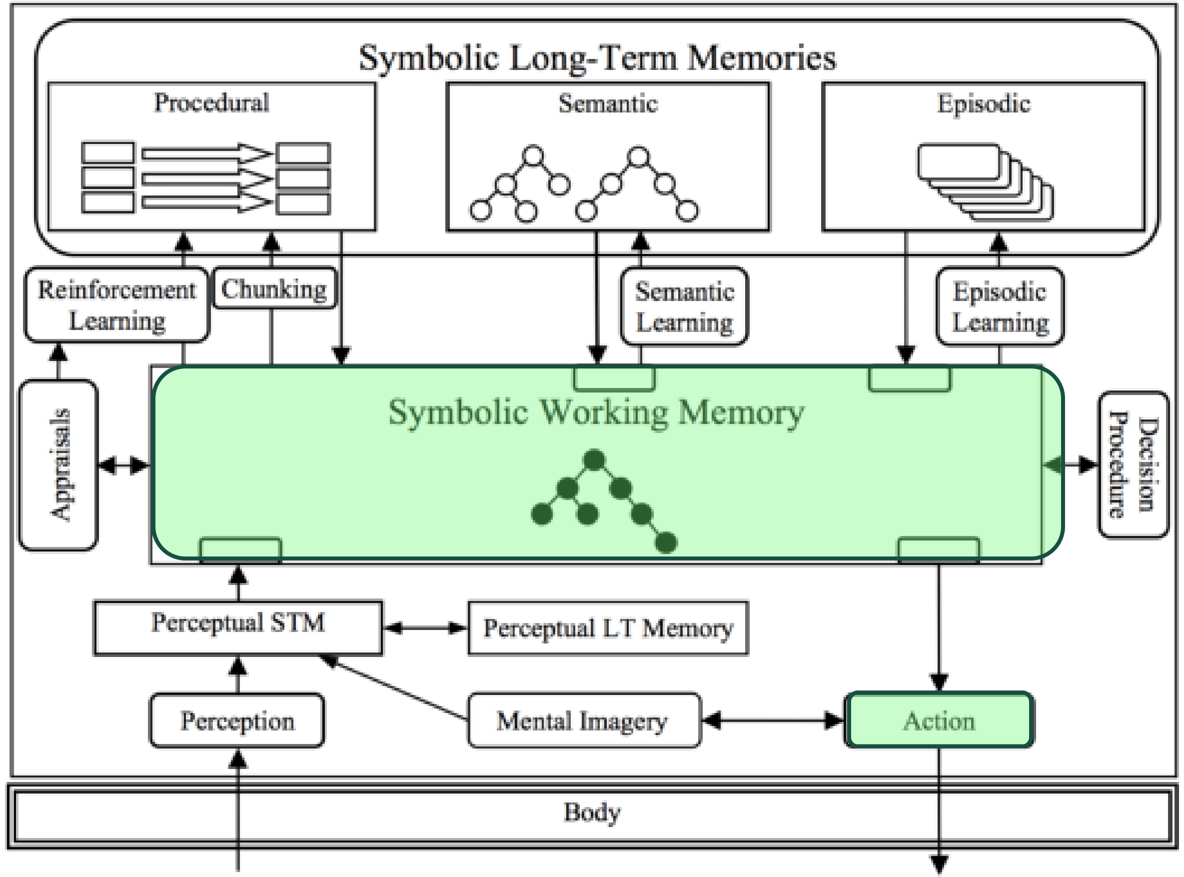
- Reinforcement learning: numeric preferences to better-performing actions
- Episodic memory: snapshots of past situations, which can be retrieved and reused in similar contexts
- Chunking (procedural learning)
  - Solved impasse: new rule in procedural memory
  - Reduce repeated reasoning



Introduction to the Soar Cognitive Architecture, Laird (2022)
https://arxiv.org/pdf/2205.03854

SOAR: Cognitive cycle - Execution

- **Input** phase: perception
- **Elaboration** phase: recognition and conceptualization
  - Interpret the situation and suggest operators
- **Decision** phase: use learned or predefined preferences to select an operator
- **Application** phase: execute **operator**
  - Change goal
  - Change belief
  - Execute action
- **Output** phase: action command



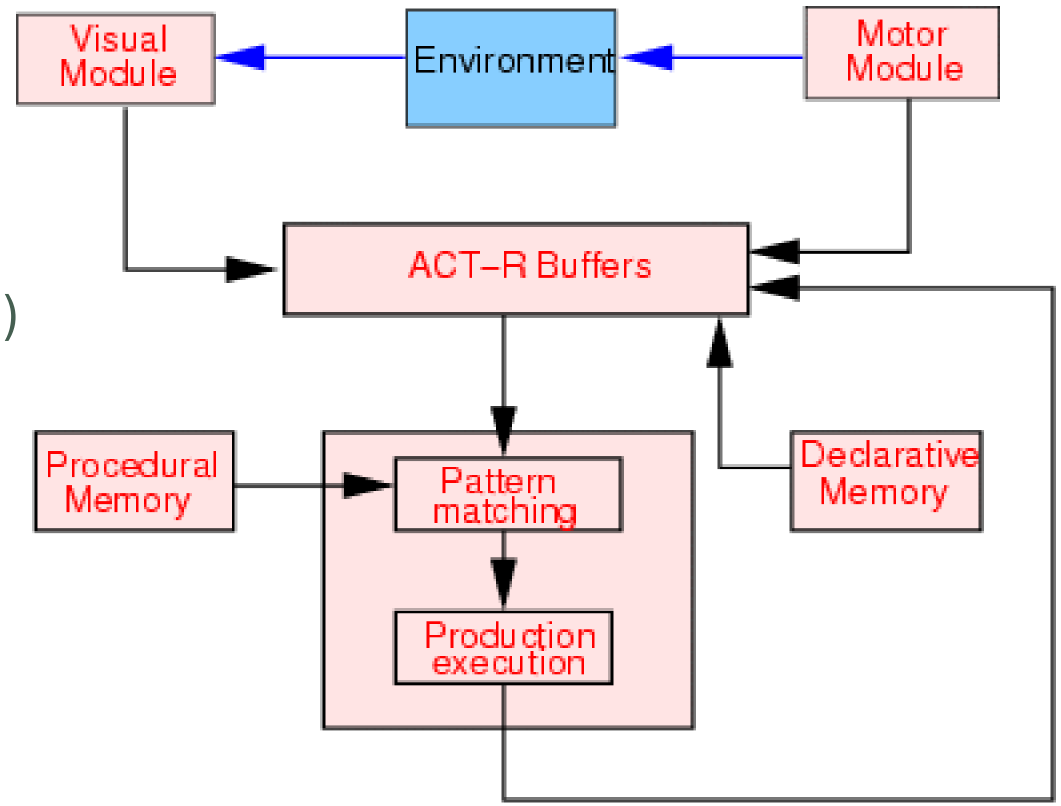
Introduction to the Soar Cognitive Architecture, Laird (2022)
https://arxiv.org/pdf/2205.03854

ROSIE: Soar agent for research

More info about Rosie: https://soar.eecs.umich.edu/rosie/

ACT-R: Adaptive Control of Thought – Rational

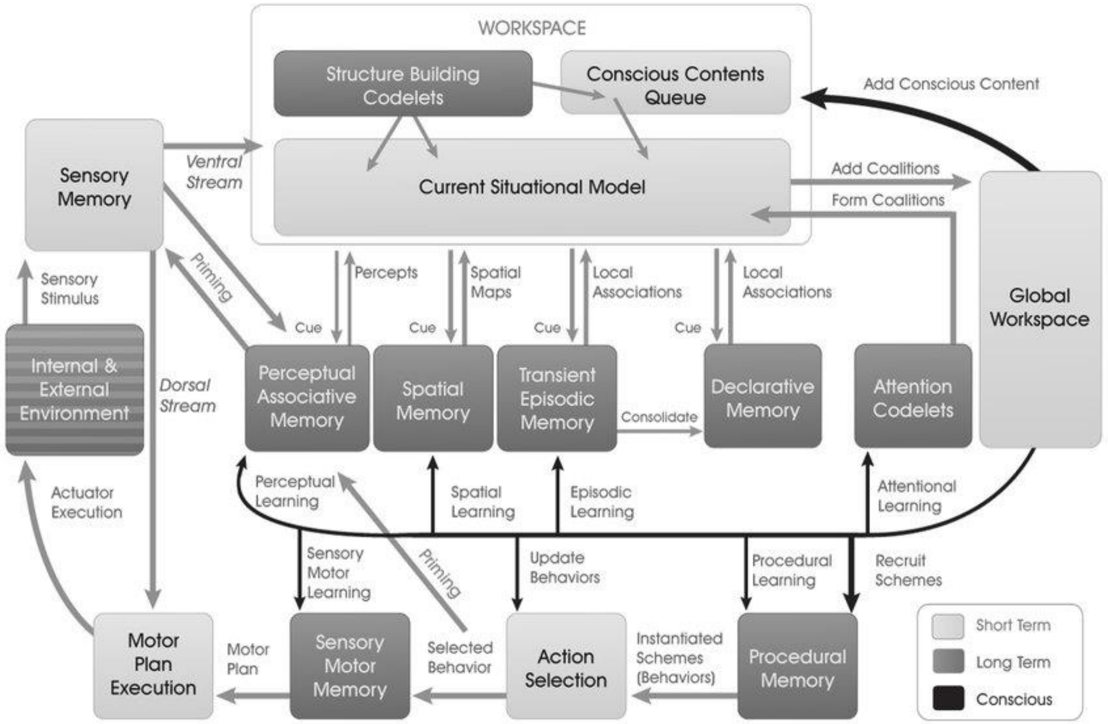
- Cognitivist architecture originally developed to simulate human experimental data
  - Maps modules into specific areas in the brain
- Memory declarative (facts) and procedural (skills)
- Use of production rules
- Perception and action managed via buffers for vision, motor, etc.
- Includes utility learning to refine rule application



ACT-R Research Group
Department of Psychology, Carnegie Mellon University
http://act-r.psy.cmu.edu/about/

LIDA: Learning Intelligent Distribution Agent

- Repeating cycles like “heartbeats of thought”:
  - Sensing: Perceive the environment
  - Attending: Broadcast salient info to the global workspace
  - Deciding: Select an action
  - Acting and Learning from the outcome
- Combines episodic, semantic, and procedural memory
- Learning every cycle: Update memories



Franklin et al. LIDA: A Systems-level Architecture for Cognition, Emotion, and Learning (2013) https://doi.org/10.1109/TAMD.2013.2277589

Part 2:
Deliberation in Robotics

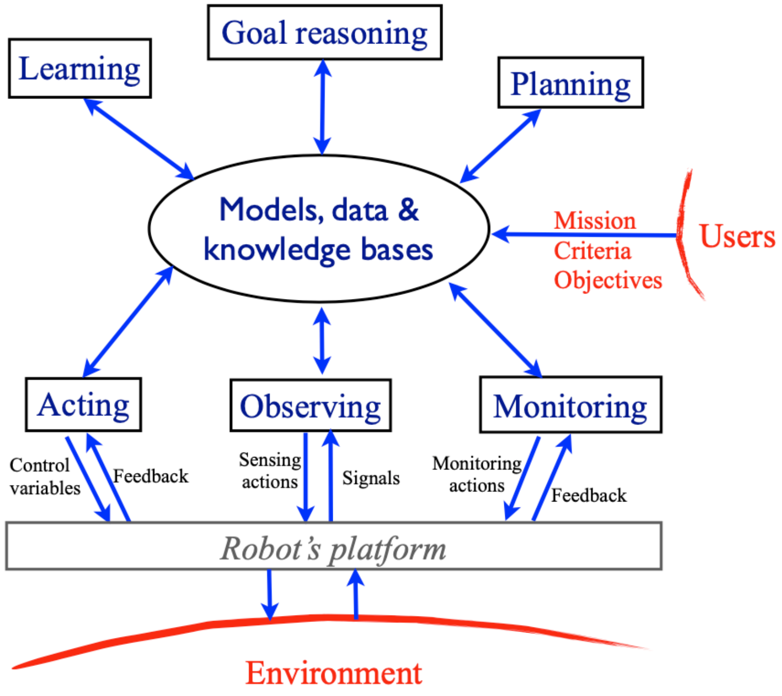
Deliberation

Deliberation is meant to endow a robotic system with extended, more adaptable and robust functionalities, as well as reduce its deployment cost.
(Ingrand & Gallab, 2017)

Deliberation

Integration of deliberative functions such as:

- Planning
- Acting
- Monitoring
- Goal reasoning
- Observing
- Learning



Félix Ingrand, Malik Ghallab, Deliberation for autonomous robots: A survey (2017) https://doi.org/10.1016/j.artint.2014.11.003

Bottleneck:

How to acquire, integrate and maintain representations to reason over them?

Part 2.1:
Deliberation in Robotics
CRAM Architecture

CORESENSE

Intelligent Robotics *Lab*

CRAM: Cognitive Robot Abstract Machine

- Hybrid cognitive architecture (symbolic & sub-symbolic representations & processes)
- Introduced by Michael Beetz in 2010 but it stills in very active development
- Designed to address robot manipulation tasks in everyday activities

EASE interdisciplinary research center at the University of Bremen, Germany

CRAM: Cognitive Robot Abstract Machine

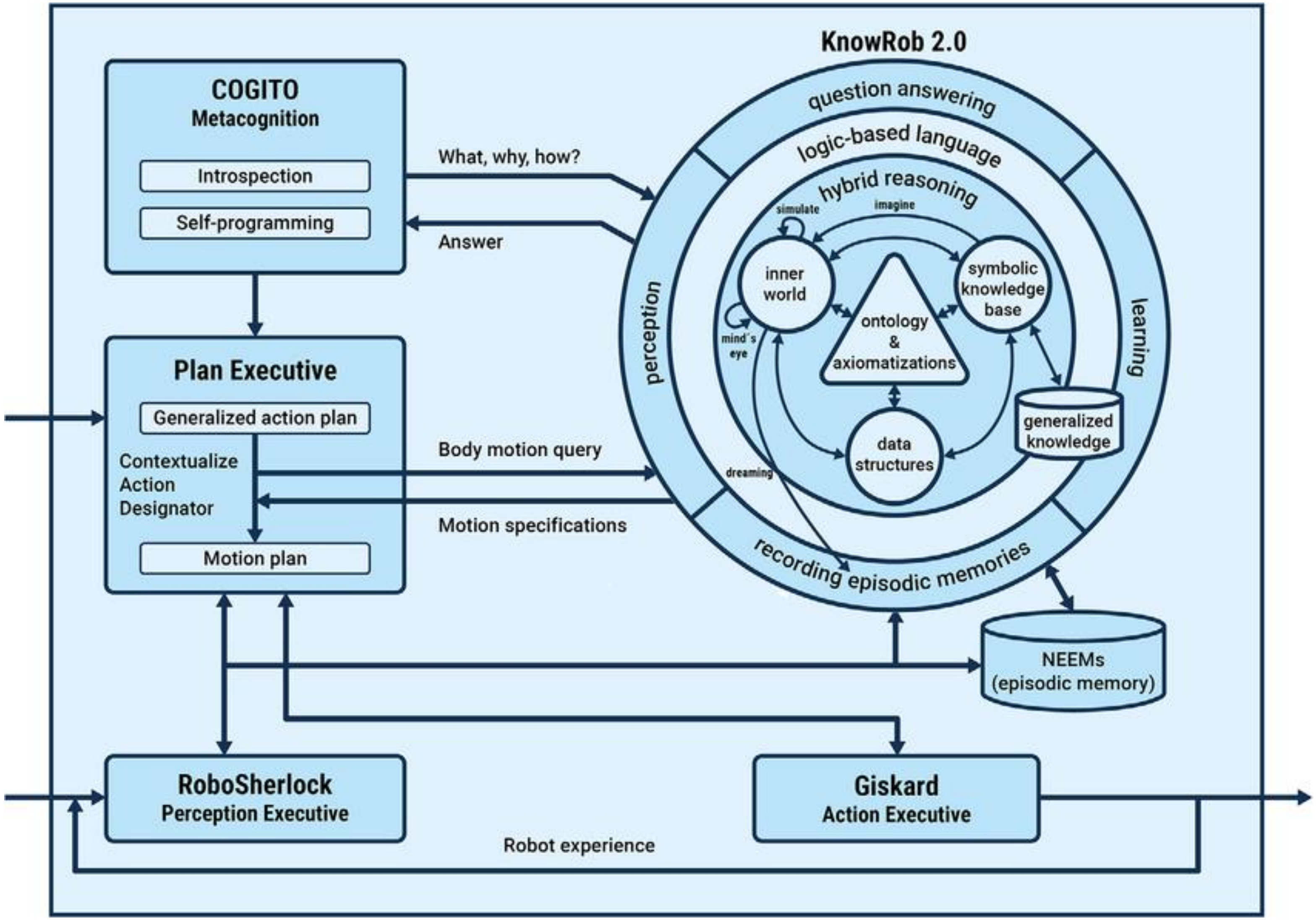
Example Goal: Make a pancake

- Get bowl
- Crack egg
- Stir
- Heat pan
- Pour mix
- Flip

CRAM handles:

- What to do next
- What tool to use
- What went wrong (e.g., “no egg found”)
- How to recover (e.g., “fetch egg from fridge”)

CRAM: Cognitive Robot Abstract Machine



Beetz et al. Robot manipulation in everyday activities with the CRAM 2.0 cognitive architecture and generalized action plans (2023) https://doi.org/10.1016/j.cogsys.2025.101375

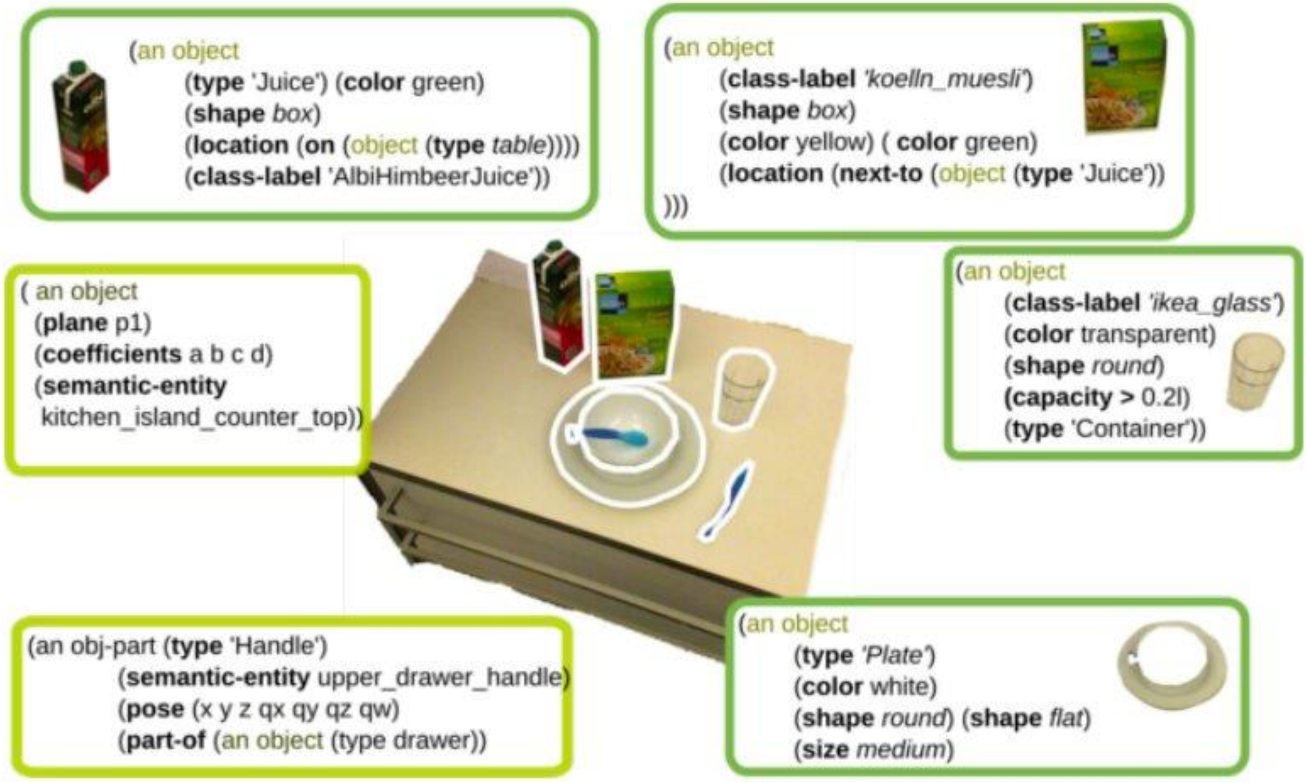
Perception: RoboSherlock

Middleware for perception

- Class/instance labels
- 6DOF positions

But also:

- Functional parts of objects
- What object is missing on a scene
- Objects contained in another object



EASE interdisciplinary research center at the University of Bremen, Germany

Perception: RoboSherlock

Uses **specialized** perception modules for different object types, environments, and tasks

- Visual detection
- Semantic knowledge reasoning
- Affordance-based inference → what can I do with this object?

Perception: RoboSherlock

- Maintains a **belief state** with virtual reality
  - Simulate what should be visible
  - Improve pose estimation
  - Save computation by guiding attention
- CRAM is shifting toward **self-supervised** perception:
  - Uses episodic memory (NEEMs) to **learn** from experience
  - Leverages internal models to generate training data automatically

Beetz et al. The CRAM Cognitive Architecture for Robot Manipulation in Everyday Activities, (2023) https://arxiv.org/pdf/2304.14119.pdf

CORESENSE

Intelligent Robotics Lab

Planning: CRAM Plan Language

- Extension of Lisp
- Specify **how** the robot should respond to:
  - Events
  - Changes in belief states
  - Detected failures

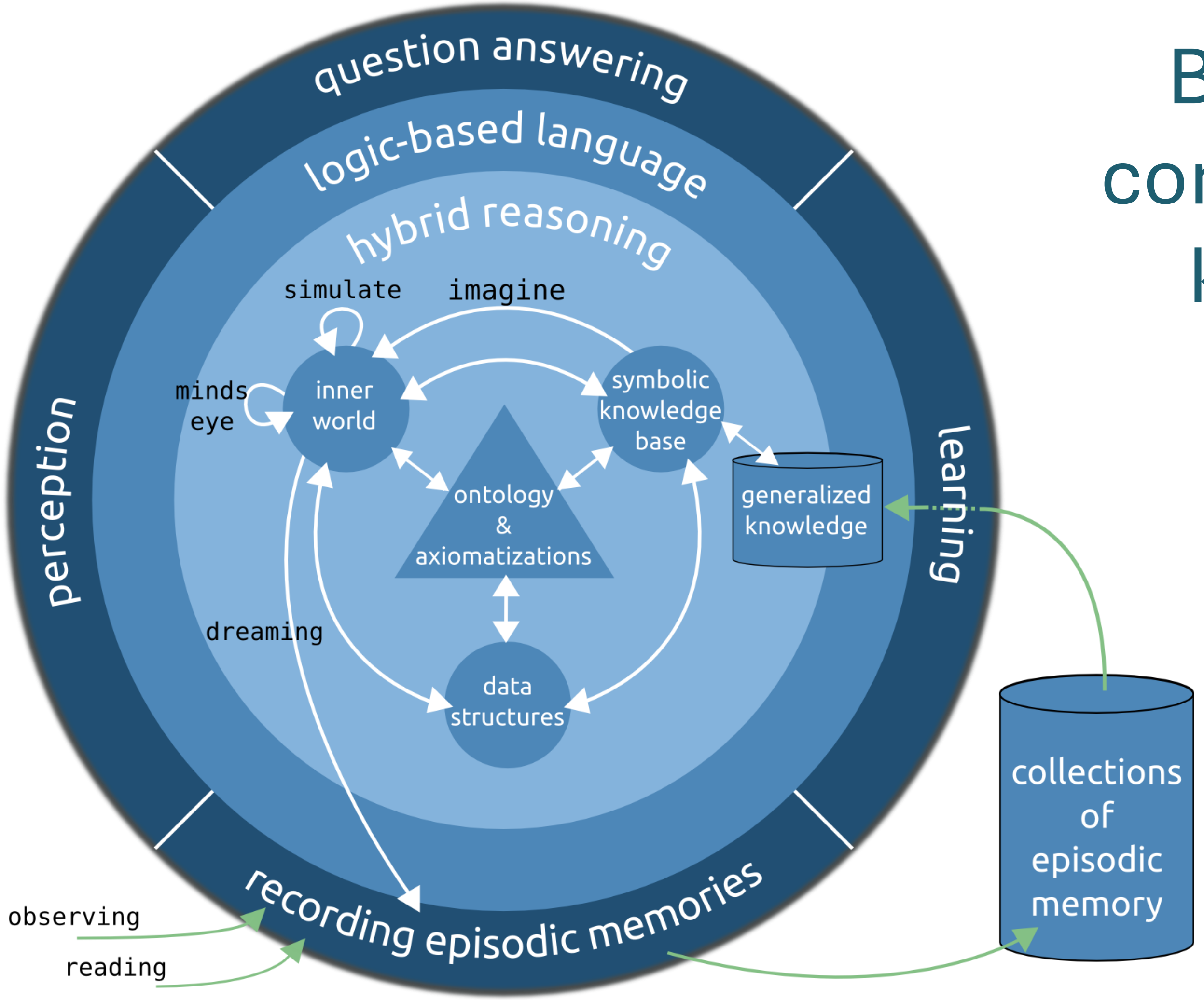
```
;; perceive package
(let ((?package-desig(perform
  (an action
      (type detecting)
       (object (an object
            (type open-box)))))))
;; pick-up the package
(perform
    (an action
        (type picking-up)
        (object ?package-desig)
        (park-arms nil)))))
```

- Supports plan **introspection**: the robot can ask itself what it was doing

Planning: CRAM Plan Language

- CRAM's execution engine **monitors plans** during execution
- If something unexpected happens (e.g. missing object), it:
  - Logs the failure
  - Adapts the plan
  - Queries to semantic knowledge to check alternatives or correct mistakes

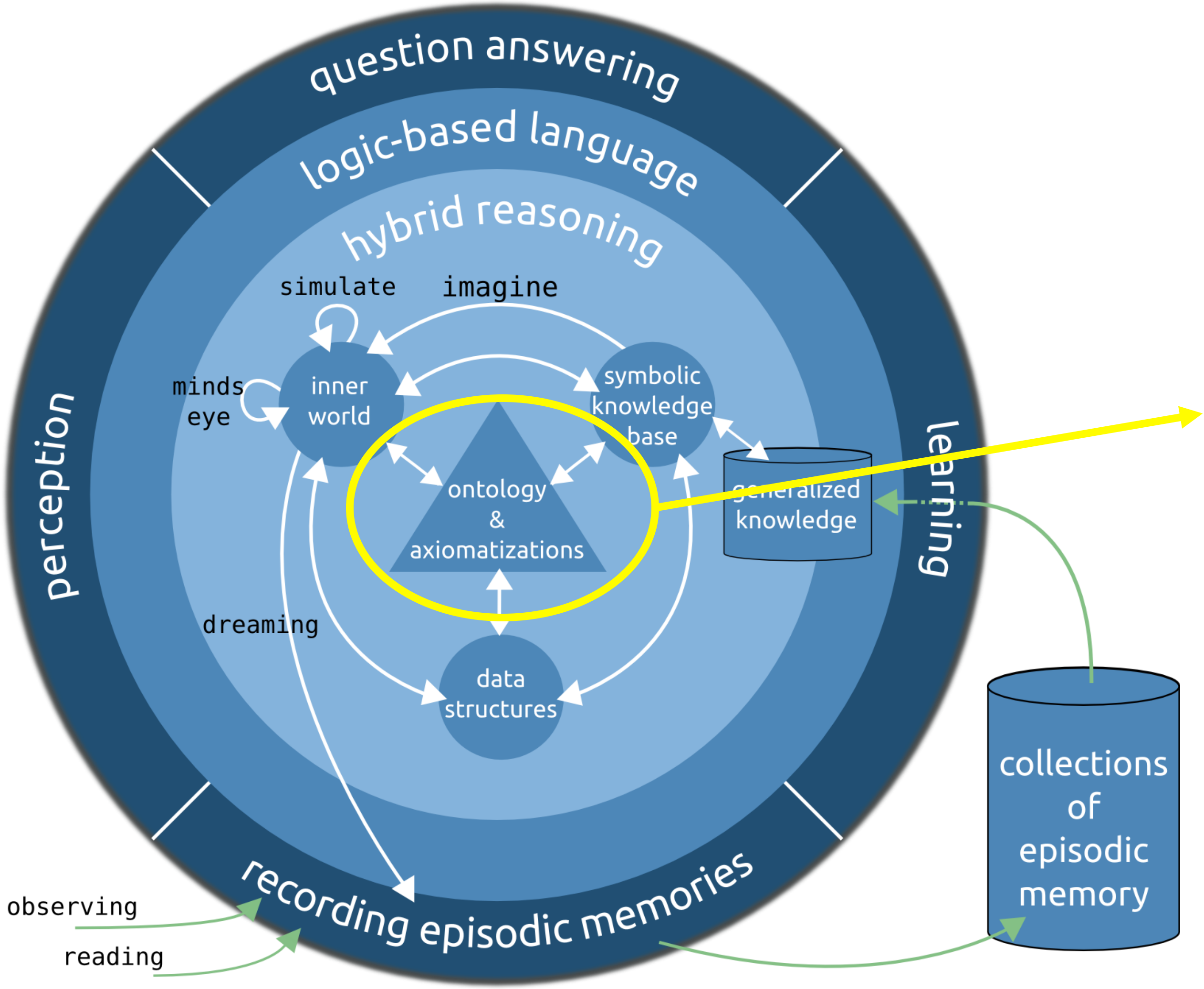
Knowledge Representation and Reasoning: KnowRob



Background common-sense knowledge

EASE interdisciplinary research center at the University of Bremen, Germany

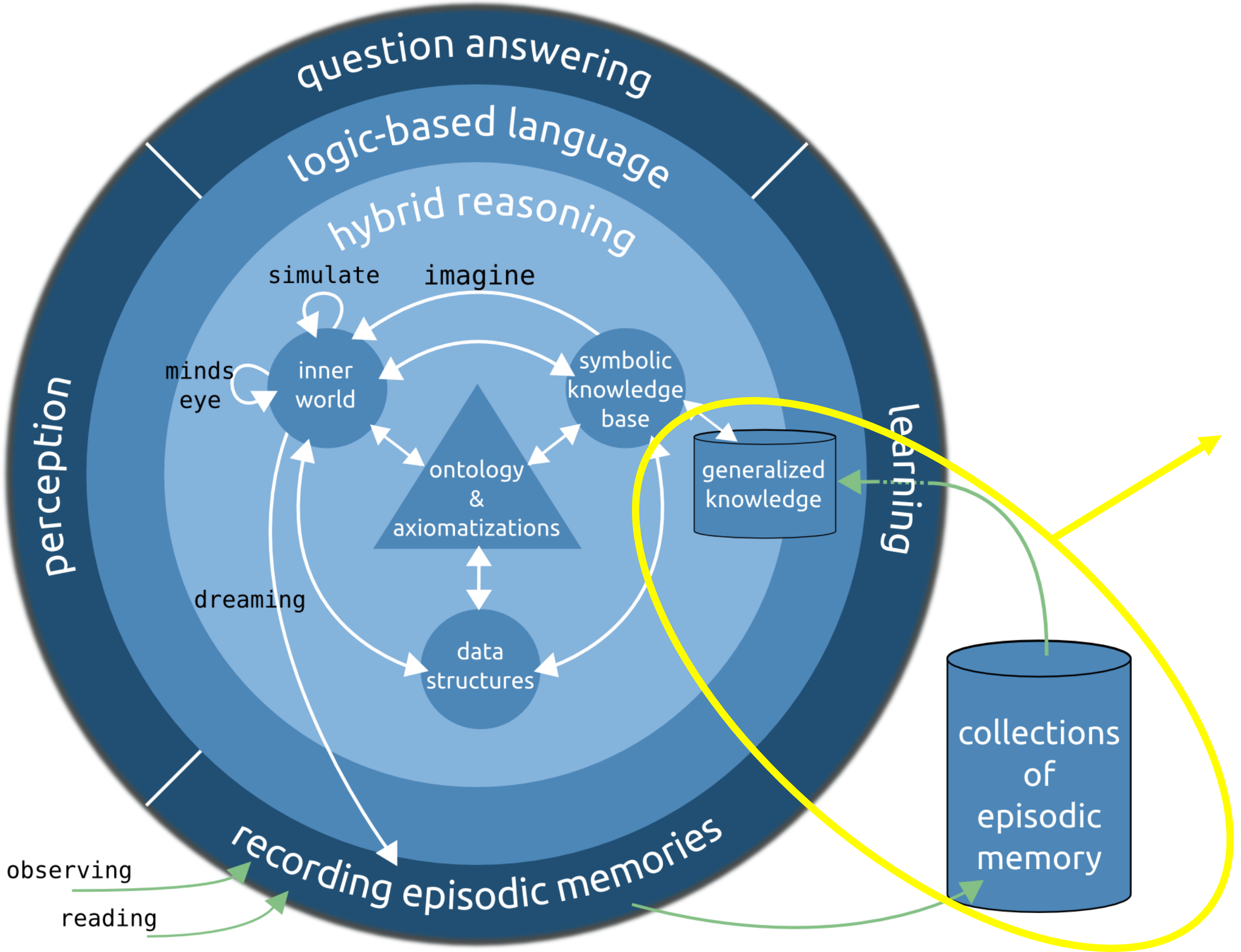
Knowledge Representation and Reasoning: KnowRob



Central set of ontologies and axiomatizations

EASE interdisciplinary research center at the University of Bremen, Germany

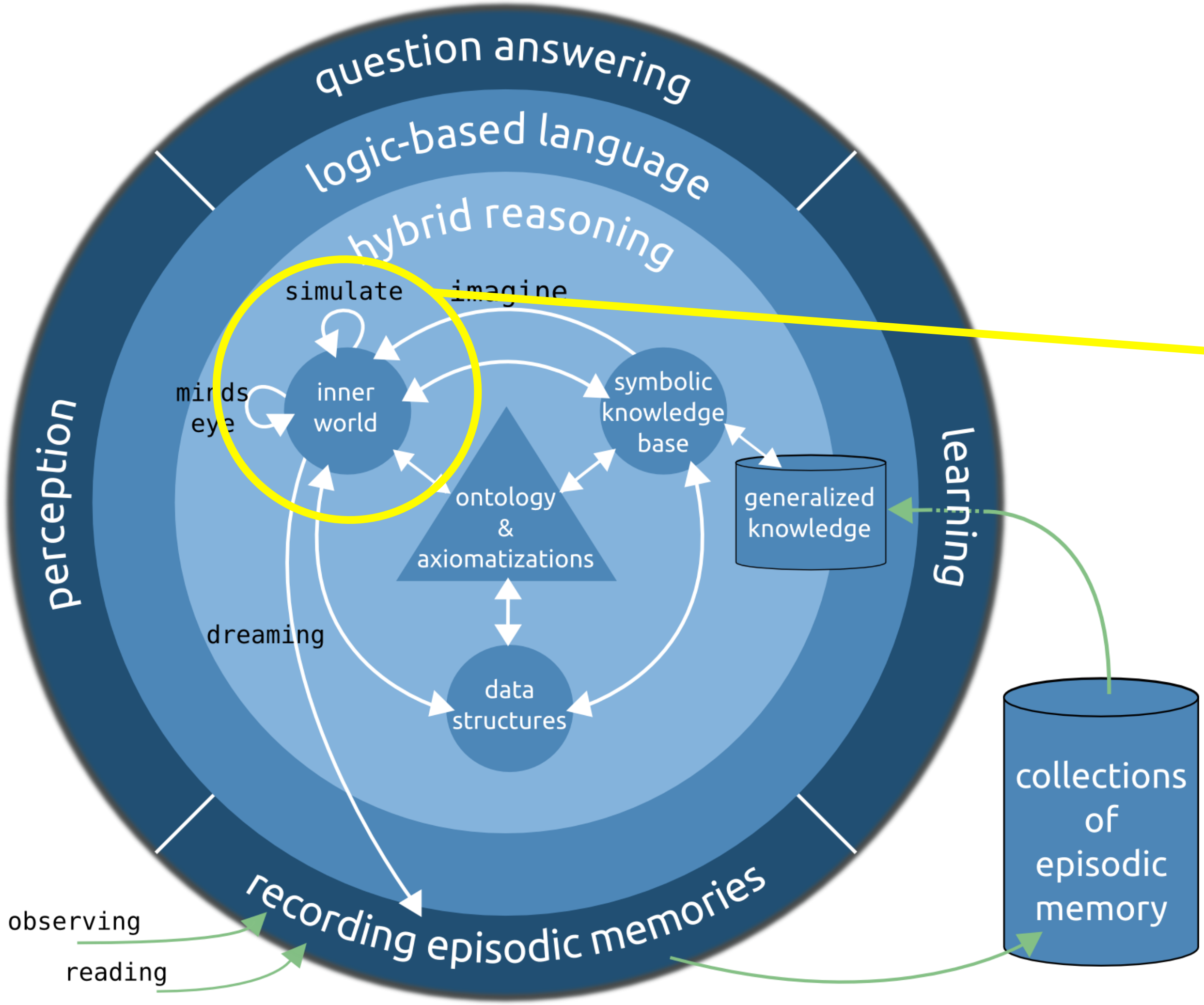
Knowledge Representation and Reasoning: KnowRob



Symbolic and generalized episodic knowledge

EASE interdisciplinary research center at the University of Bremen, Germany

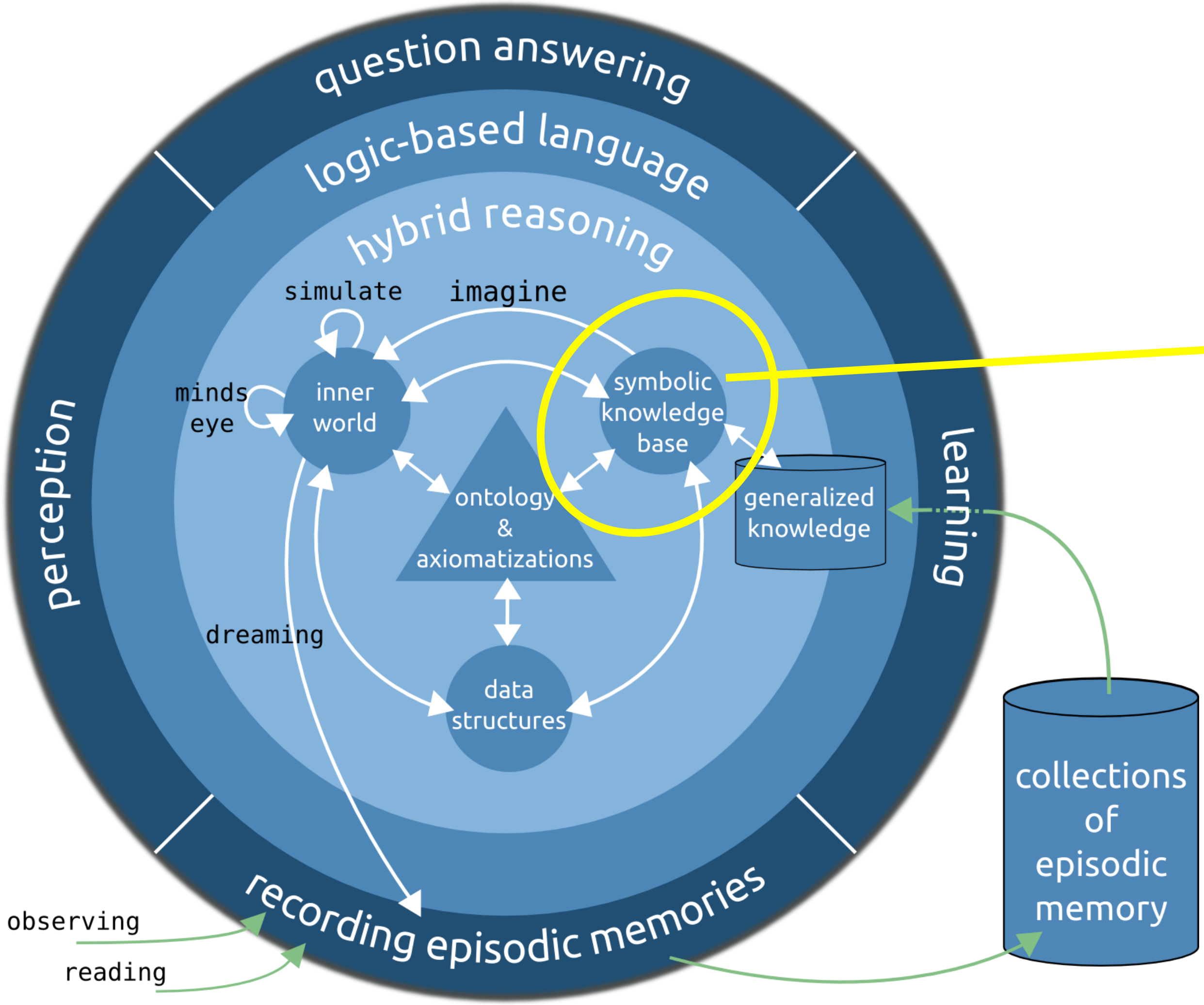
Knowledge Representation and Reasoning: KnowRob



Abstract knowledge from physics simulator

EASE interdisciplinary research center at the University of Bremen, Germany

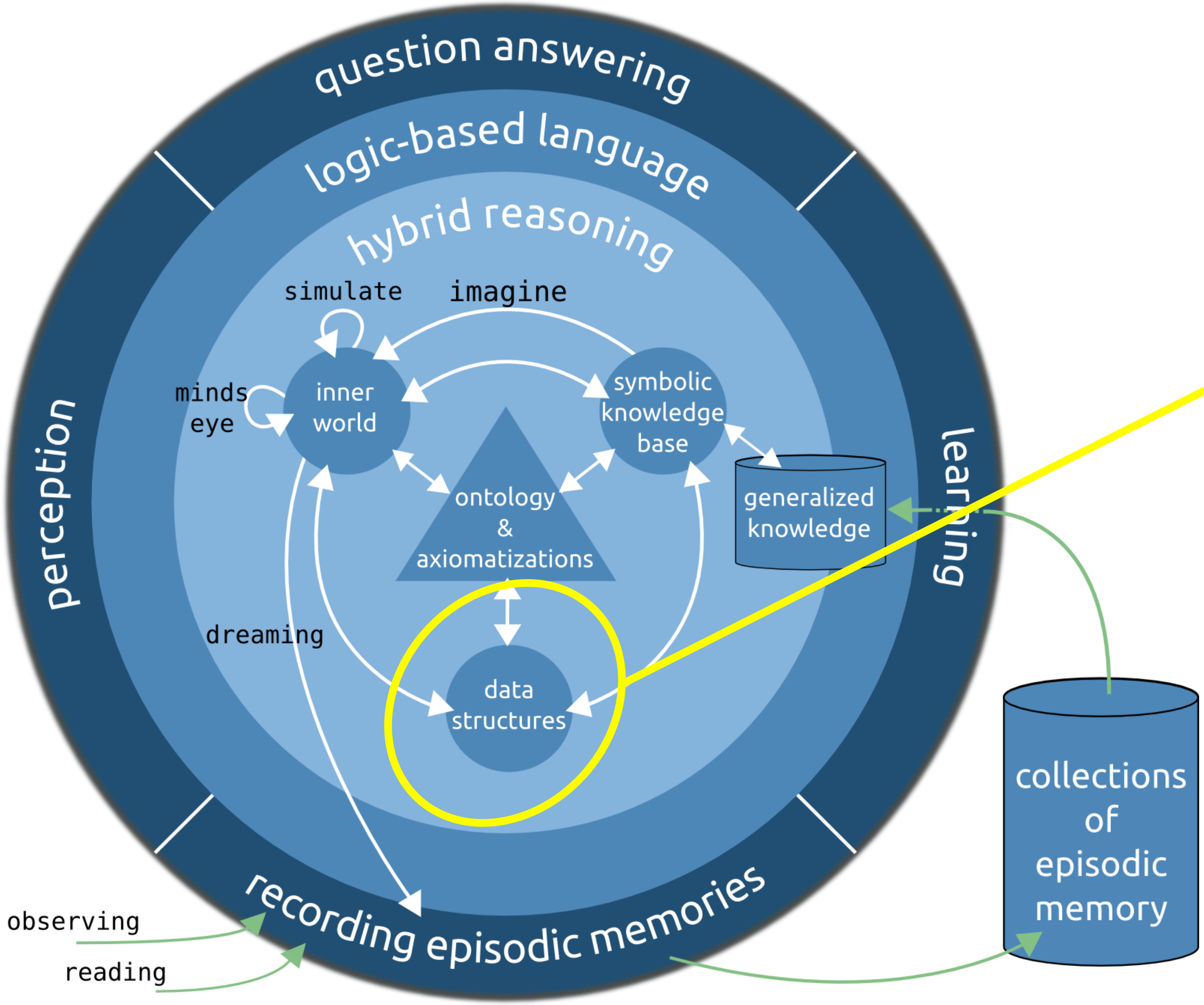
Knowledge Representation and Reasoning: KnowRob



Logic KB with rules for sensor and action data, logical axioms and inference rules

EASE interdisciplinary research center at the University of Bremen, Germany

Knowledge Representation and Reasoning: KnowRob



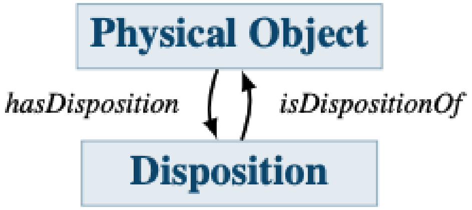
Virtual KB to parameterize motion control and path planning

EASE interdisciplinary research center at the University of Bremen, Germany

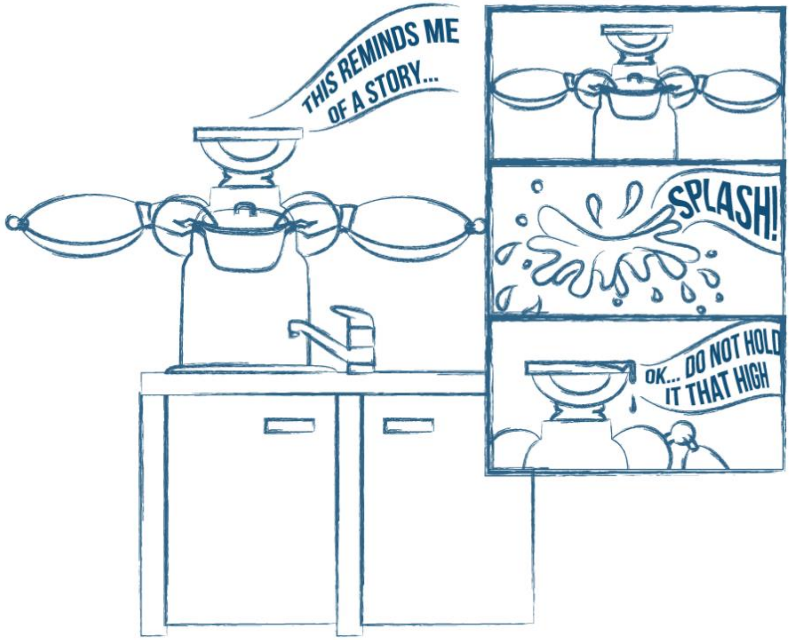
NEEMs: Narrative Enabled Episodic Memories

Learn from experience and update KB

Intent	To represent what kinds of interactions an object can participate in.
Competency Questions	*What can this object be used for? Can this object interact with others in a particular way?*
Defined in	SOMA.owl



Expression	**Meaning**
has_disposition(x,y)	$y \in \mathscr{A}$ is a disposition of $x \in \mathscr{A}$

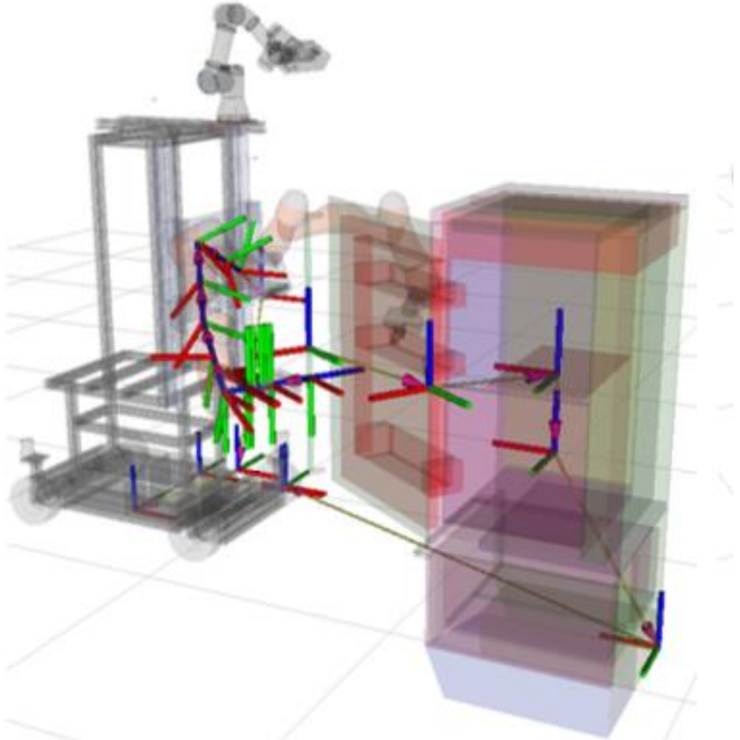


EASE interdisciplinary research center at the University of Bremen, Germany

Motor execution: Giskard

- Calculates body movements based on idealized, abstract robot capability models
- Most motion learned by reinforcement learning (NEEMs)
- Active research:
  - Tactile-based manipulation
  - Optimization for task force and touch control (e.g., slicing bread)

Example Goal:
Keep holding the door and move it according to its joint model



Beetz et al. The CRAM Cognitive Architecture for Robot Manipulation in Everyday Activities, (2023) https://arxiv.org/pdf/2304.14119.pdf

Metacognition: COGITO

Example Question:
"Can the action goal be achieved?" or "Did the action fail because the robot didn't see the object?"

- **Reason** about system performance and **adapt** to improve its effectiveness
- Queries, their responses, and the success or failure of actions **logged** during execution
- Fully **integrated** with CRAM Planning: understand subplans and its effects
- Use of KnowRob to **answer** "why" questions
- NEEMs to establish causal relationships (motion → environmental change)
  - Reprogram plans, e.g., close a door pushing with an elbow

CRAM: Limitations

- Steep learning curve: Lisp and Prolog/OWL
- Plan **adaptation** is pre-modeled
- KnowRob's logic-based reasoning can become **computationally expensive** for large ontologies or high-frequency queries

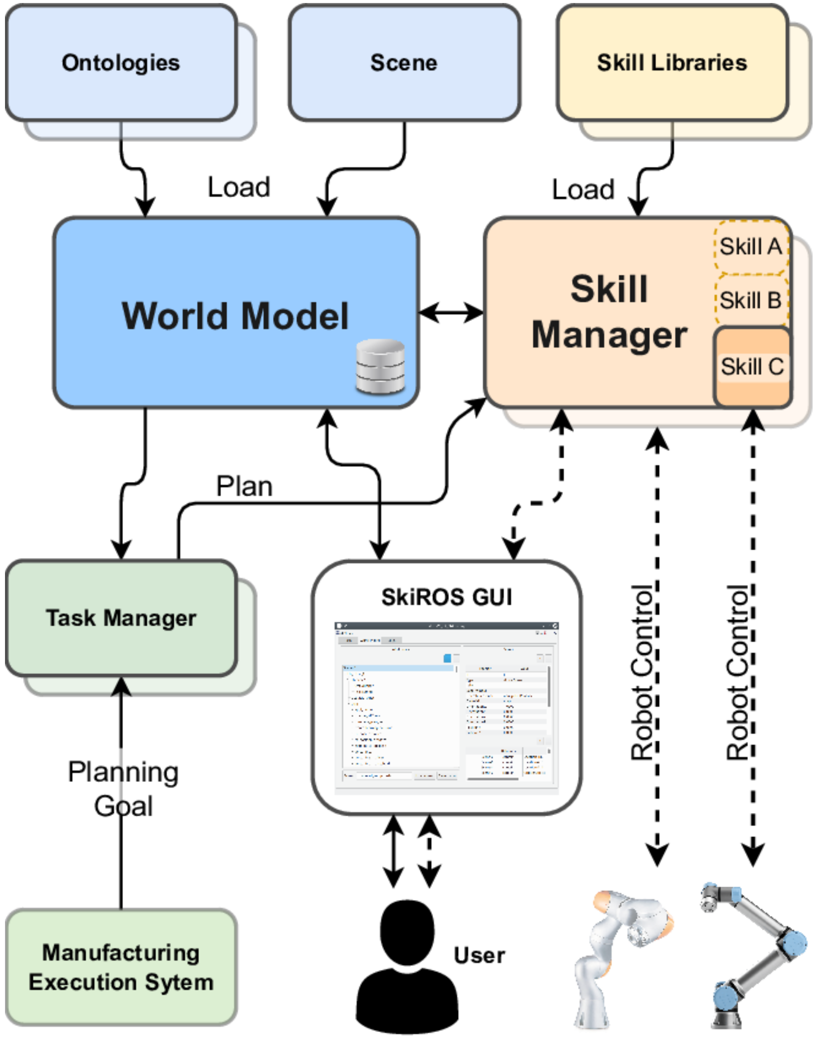
Part 2.2:

Deliberation in Robotics

SkiROS

SkiROS2: Skill-based robot control platform

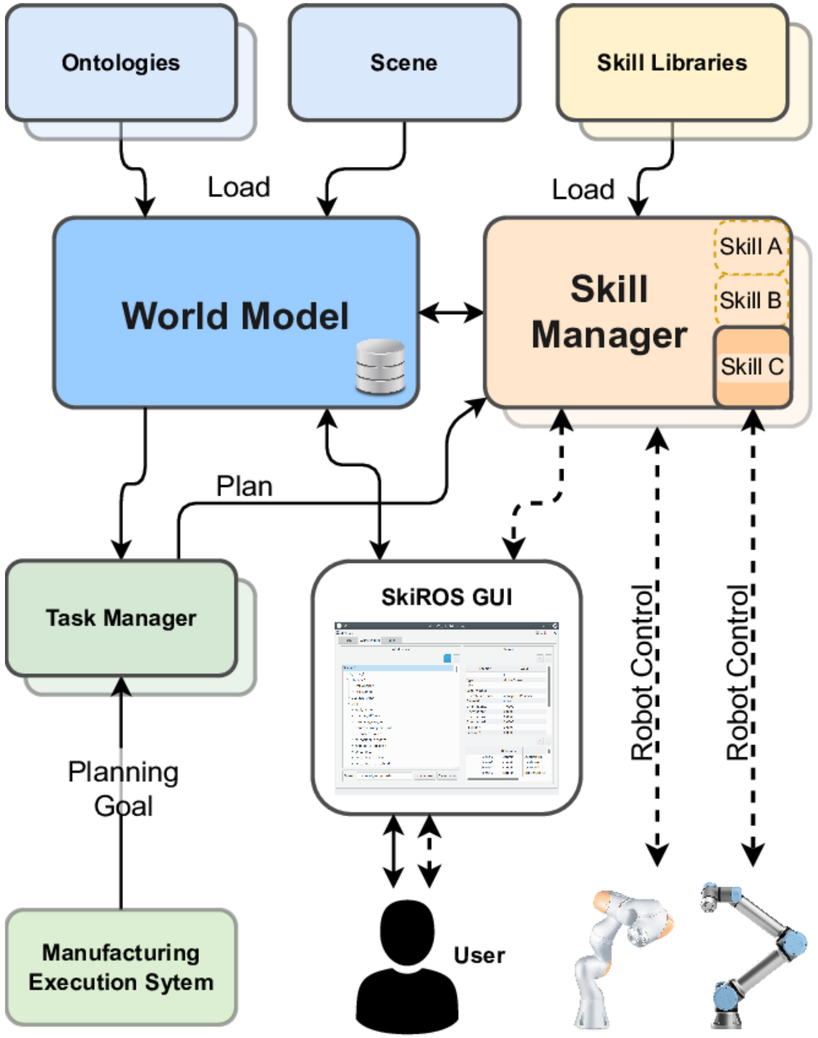
- Engineering approach
- Objective: handle system complexity in intelligent systems performing industrial tasks
- Coordination of partial solutions and interoperability across different robots



Mayr et al., *SkiROS2: A skill-based Robot Control Platform for ROS, (2023)*
https://doi.org/10.1109/IROS55552.2023.10342216

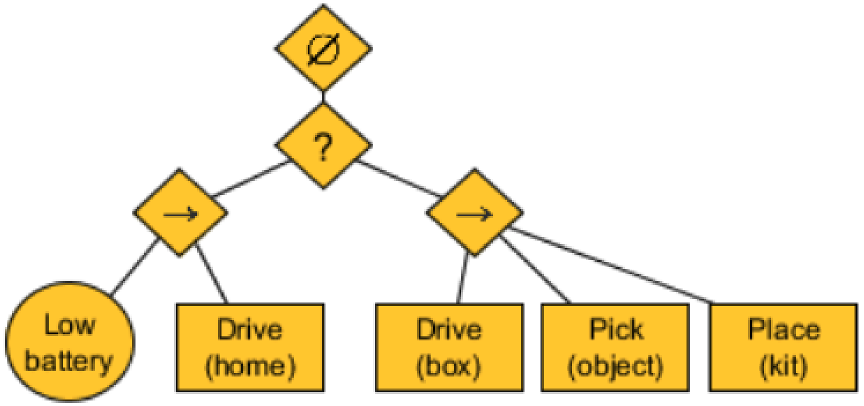
SkiROS2: Planning

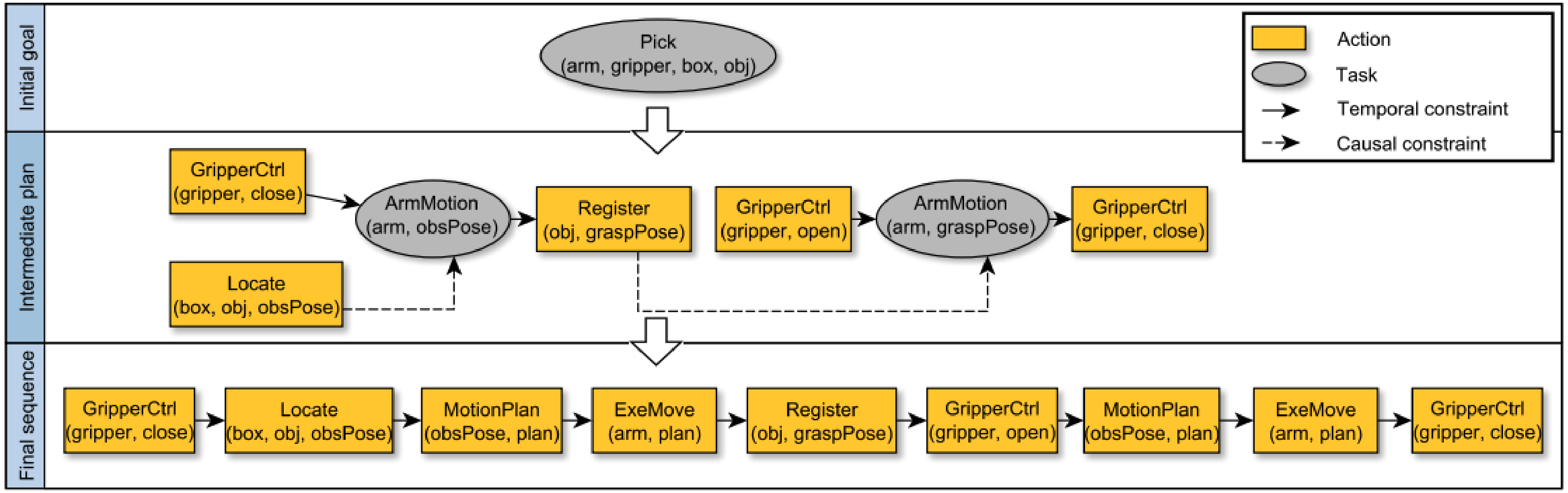
- Task manager: PDDL to find skill sequence
- Behaviour tree: directed acyclic graph → execution of actions
  - Link nodes with conditions and logical relations (executed in sequence, alternative or in parallel)
  - Actions return success, failure or running
- Extended behaviour trees (eBT): add pre and post condition nodes → hierarchical task network (HTN)



Mayr et al., *SkiROS2: A skill-based Robot Control Platform for ROS*, (2023) *https://doi.org/10.1109/IROS55552.2023.10342216*

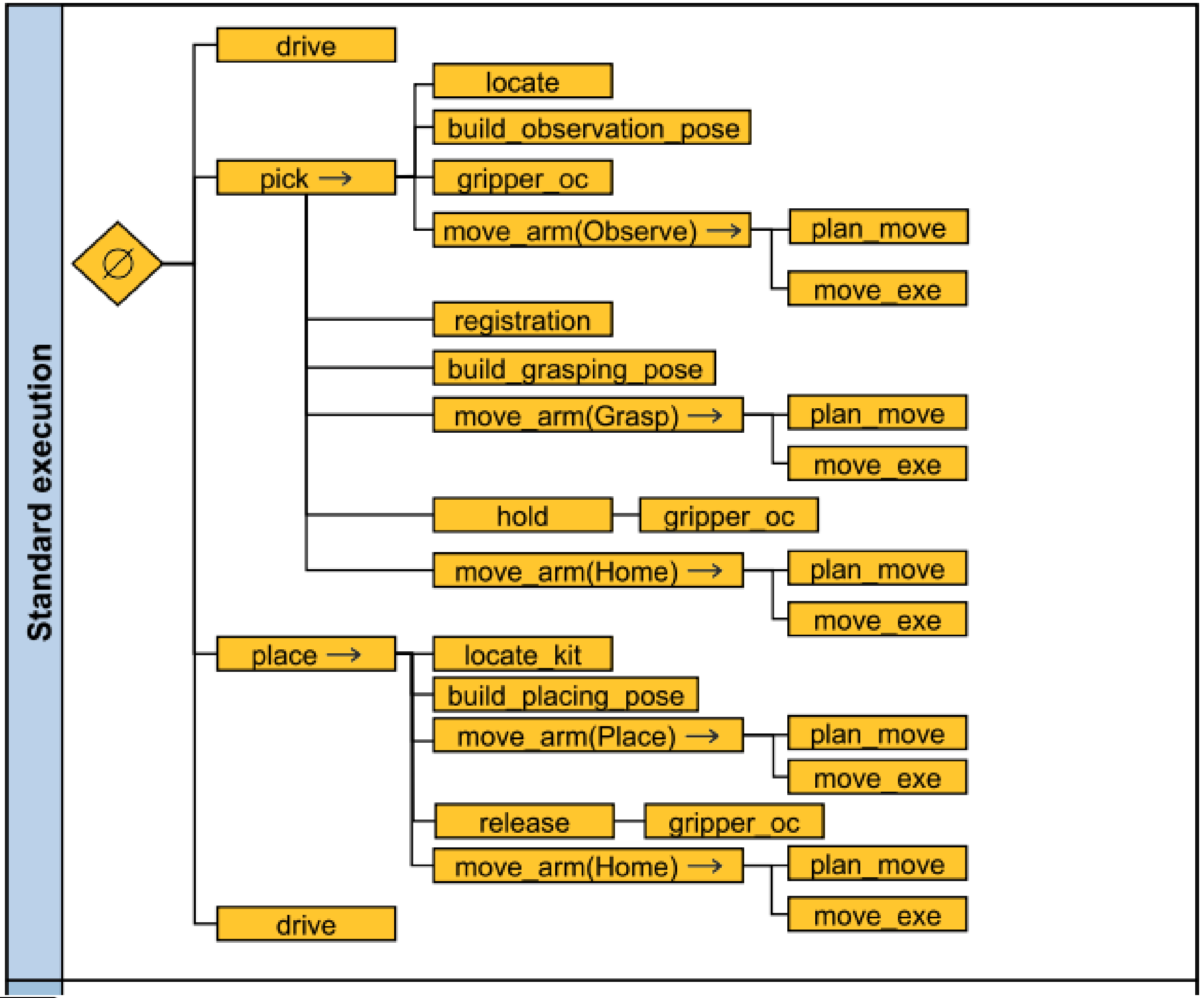
SkiROS2: Planning – BT + HTN

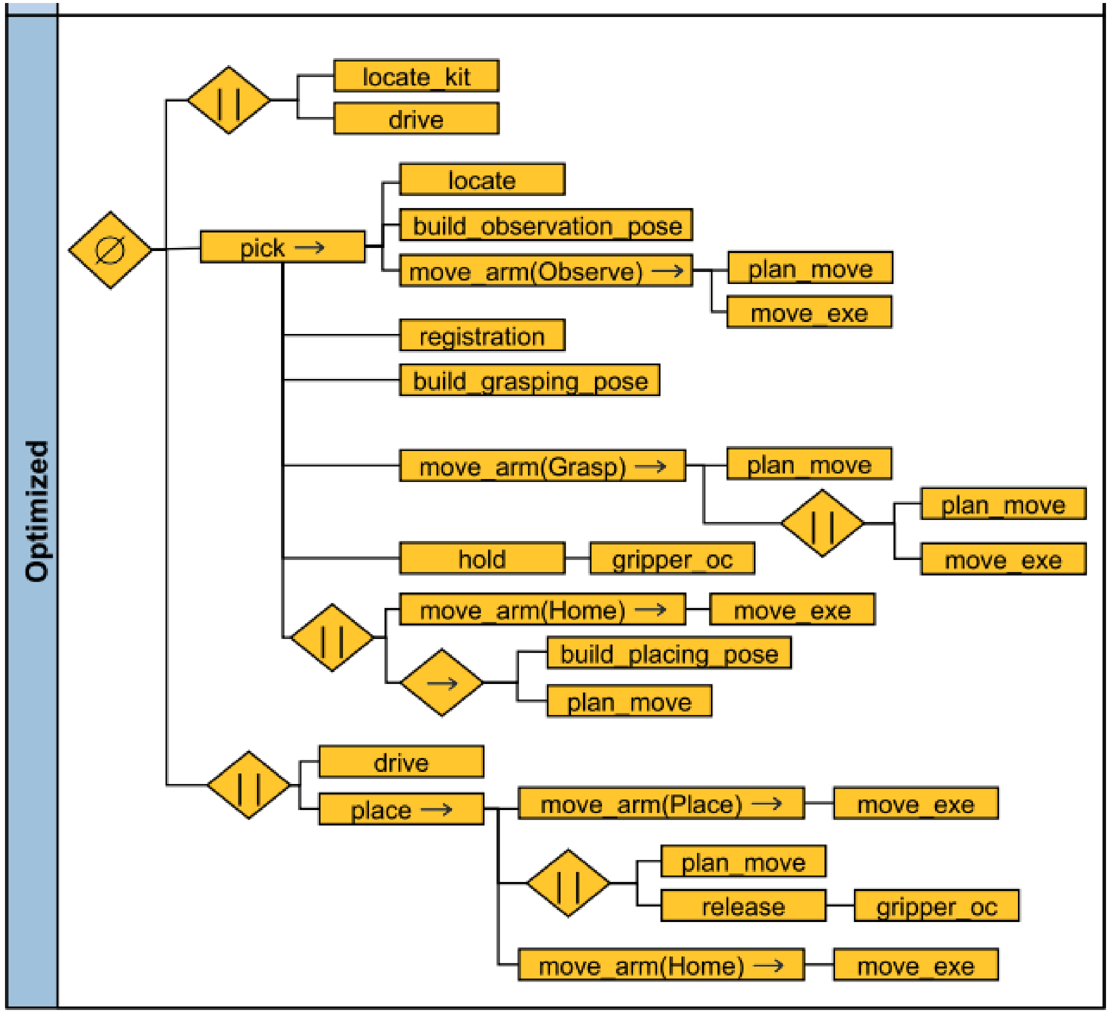




Rovida et al., Extended behavior trees for quick definition of flexible robotic tasks. (2017)
http://doi.org/10.1109/iros.2017.8206598

SkiROS2: Planning – eBT





Rovida et al., Extended behavior trees for quick definition of flexible robotic tasks. (2017)
http://doi.org/10.1109/iros.2017.8206598

SkiROS2: Knowledge representation

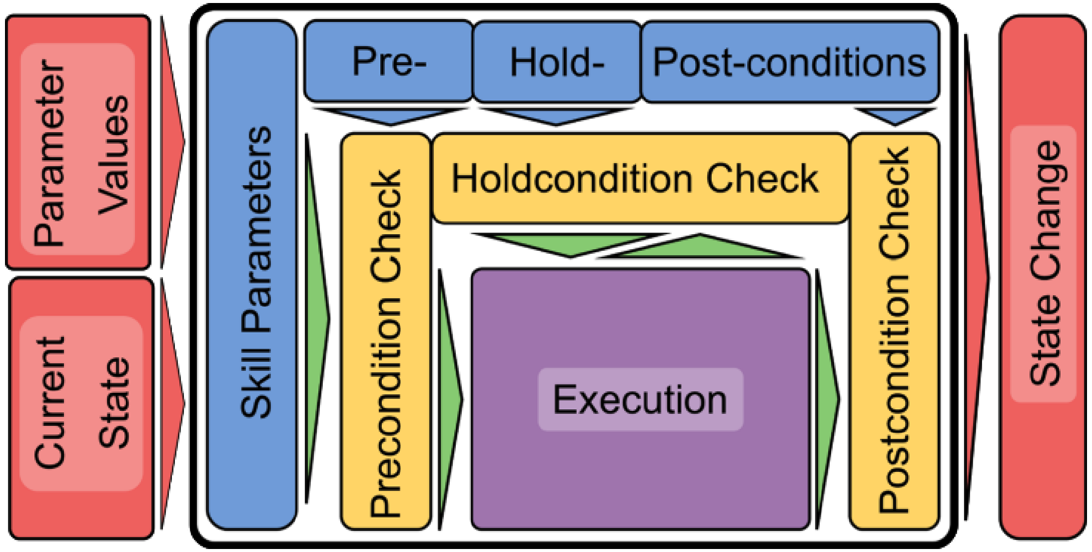
- Stores knowledge in an RDF graph (OWL)
  - Ontologies (Core Ontology for Robotics and Automation)
  - Concepts
  - Properties
  - Relations
- Enables reasoning and planning
- World model shared across robots

Subject	Predicate	Object
skiros:Container	rdfs:subclassOf	skiros:Location
skiros:DriverAddress	rdfs:subPropertyOf	skiros:DeviceProperty
skiros:Scene-0	skiros:contains	skiros:Location-1
skiros:Robot-2	skiros:at	skiros:Location-1

Mayr et al., SkiROS2: A skill-based Robot Control Platform for ROS, (2023)
https://doi.org/10.1109/IROS55552.2023.10342216

SkiROS2: Skills

- Skill as parameter procedure that transform a state
- A skill manager per robot
- Atomic and compound skills (eBTs)
- Semantic description
  - Parameters (required, inferred, optional)
  - Pre-, hold-, post-conditions



Mayr et al., SkiROS2: A skill-based Robot Control Platform for ROS, (2023)
https://doi.org/10.1109/IROS55552.2023.10342216

SkiROS2: Limitations

- **Static** knowledge base
- Skill representations are modular and reusable, but **hardcoded** in plugins
- No native support for self-monitoring, metareasoning or advance perception (e.g., reflection on failed plans, uncertainty handling)
- Does **not introspect** about why a failure happened or how to revise its strategy

Part 2.3:
Deliberation in Robotics
SysSelf

CORESENSE

Intelligent Robotics *Lab*

Our approach

- Capturing knowledge:
  - Represent and integrate expert and domain-specific knowledge
  - This enables the robot to directly use sophisticated, pre-existing intelligence embedded within its architecture during task execution
- Supporting metacognitive capabilities:
  - Incorporate mechanisms for representing knowledge about their own internal states and capabilities

"How can we enhance autonomous robots' self-awareness from a systemic perspective to make them more robust?"

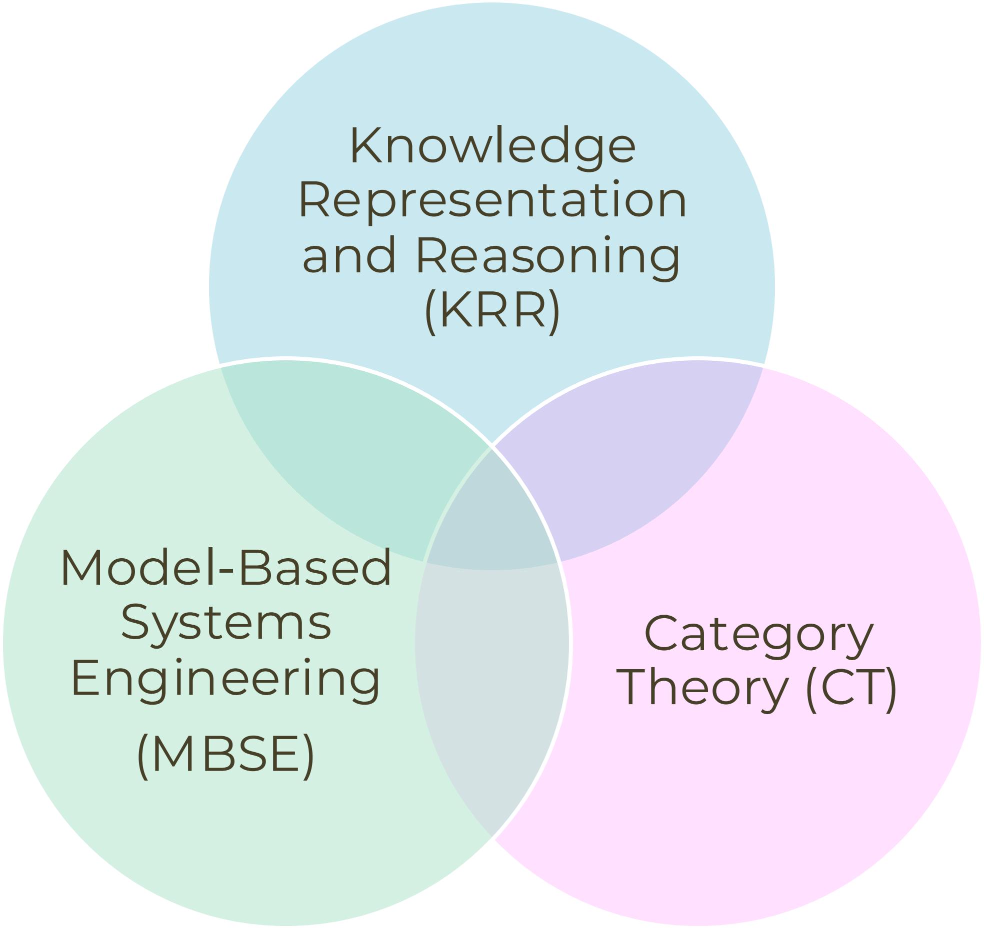
Requirements

- Capture system structure
- Reuse existing definitions
- Value-oriented
- Applicable to a variety of systems
- Use declarative formal language
- Runtime executable

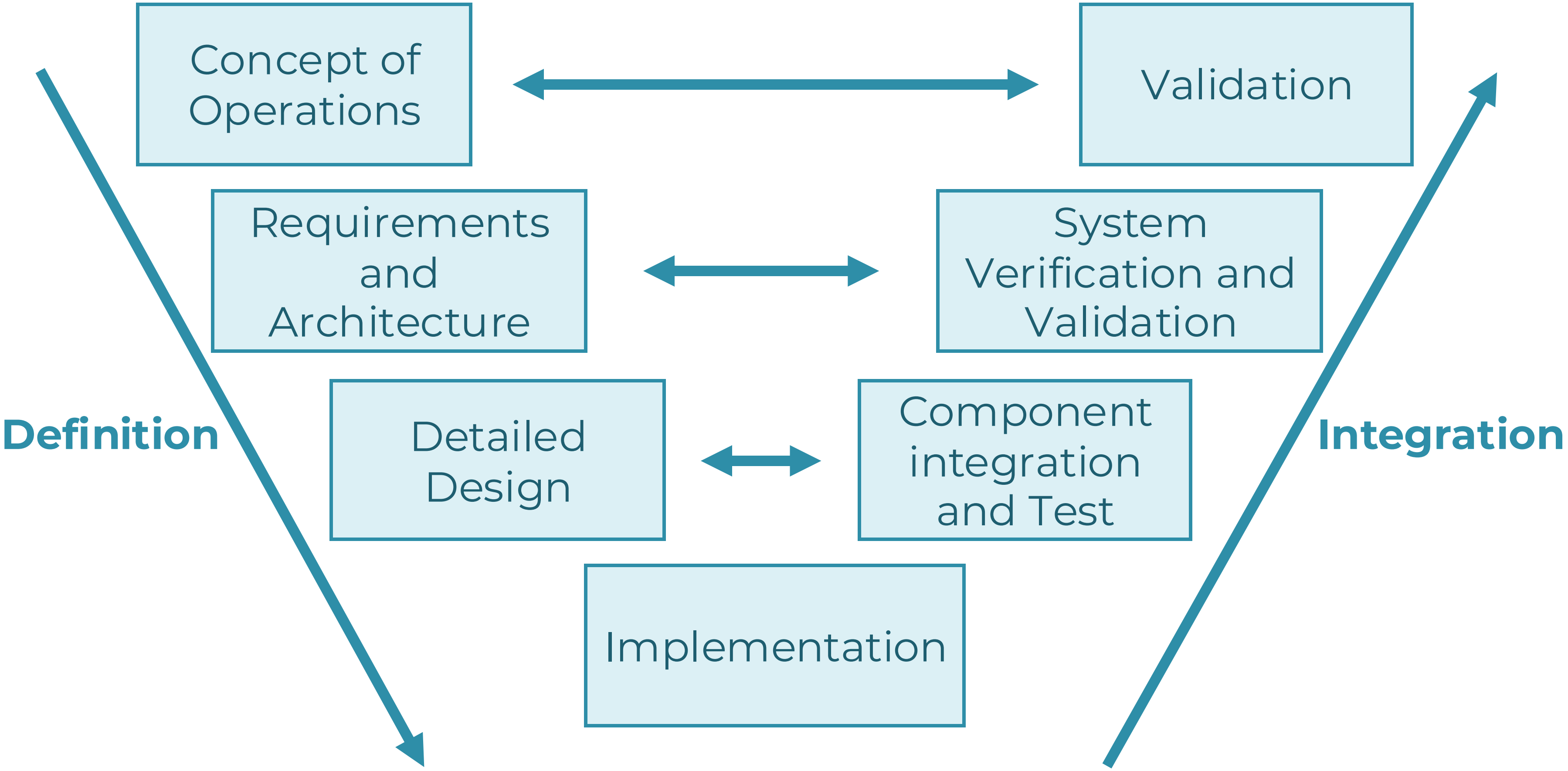
MBSE

CT + OWL

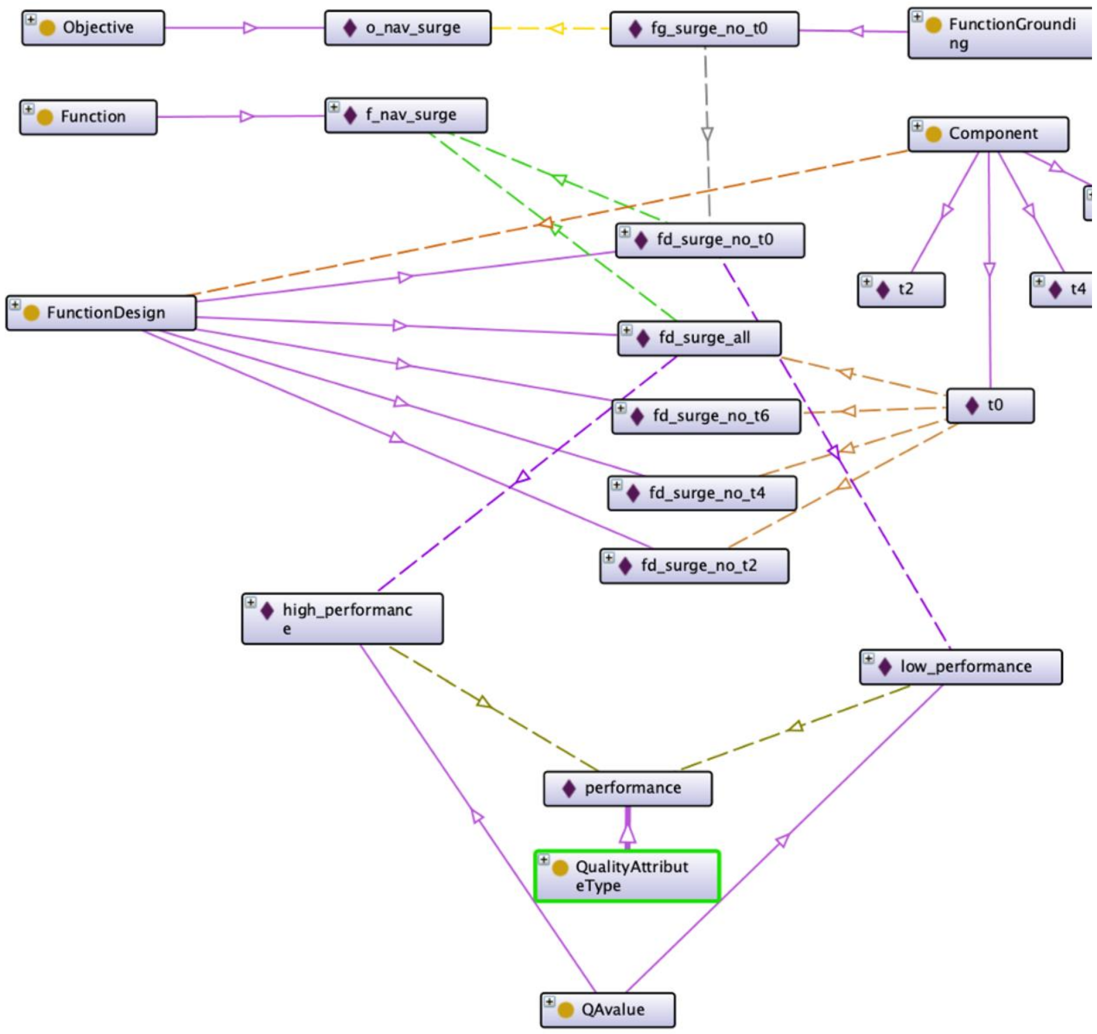
Involved domains



Model-Based Systems Engineering

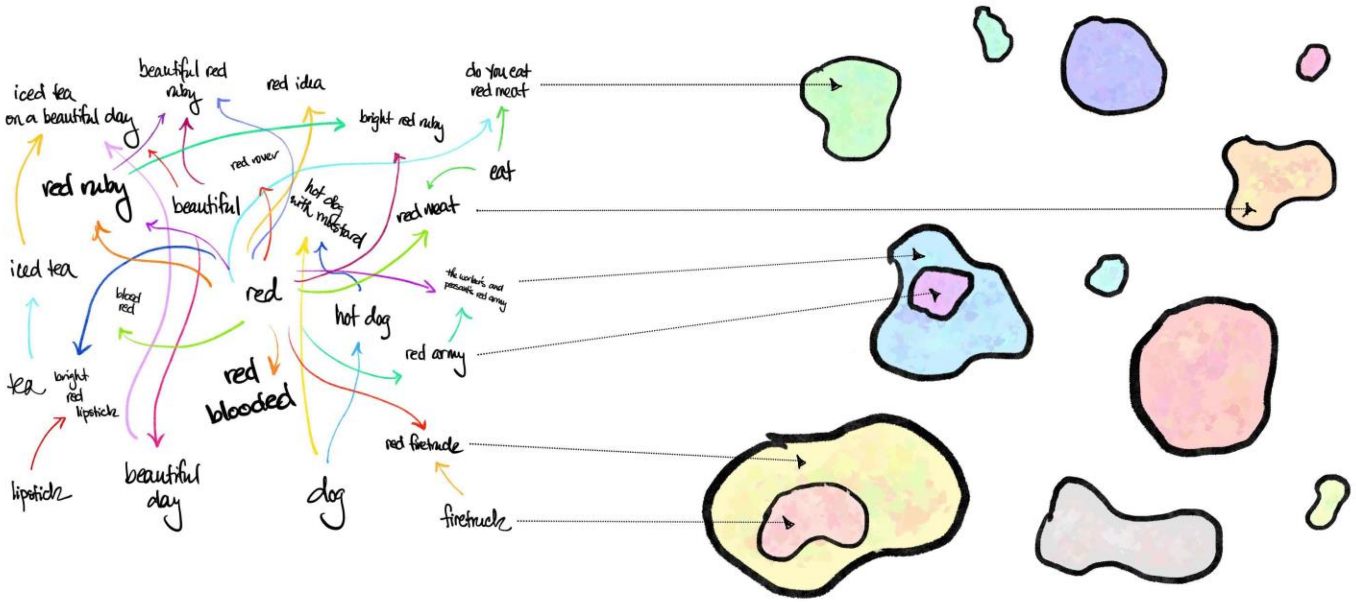


Knowledge Representation and Reasoning



Category Theory

- General theory of mathematical structures
- Compositionality
- Consistency



Bradley et al. Math3ma blog: https://www.math3ma.com/blog/language-statistics-category-theory-part-3

Category Theory: Basic elements

- Category:
  - Objects
  - Morphisms: map between objects
  - Binary operator: composition of morphisms
- Functor: Map between categories
- Natural transformations: Map between functors

Category Theory: Basic elements

$$
\begin{tikzcd}
A \arrow[r, "g"] \arrow[d, "f"'] & Y \arrow[d, "f'"] \arrow[ddr, bend left, "l_y"] & \\
X \arrow[r, "g'"'] \arrow[drr, bend right, "l_x"'] & \boxed{X_{+A}Y} \arrow[dr, "u"] & \\
 & & \boxed{T}
\end{tikzcd}
$$

Category Theory: Basic elements

Specification ⟵ **System** ⟶ Operation

Category Theory: Equivalence

- Morphisms, functors, natural transformations
- Yoneda lemma:
  - Equivalence of two objects in a category from relationships
  - Formal representation of system design alternatives

Metamodel

- Designed to model-based adaptation to robustly deliver the expected value

- Main concepts:
  - Capability
  - Component
  - Goal
  - Value
  - Stakeholder
  - Metrics:
    - MOE
    - MOP
    - TPM
  - Constraint
  - Interface

Metamodel: Categories

- Component:
  - Objects: motors, sensors, controllers, etc.
  - Morphisms: dependencies and interfaces between components
- Capability:
  - Objects: sense, move, decide, plan, etc.
  - Morphisms: dependencies and synergies
- Goal:
  - Objects: desired position, extract quantity of mineral, etc.
  - Morphisms: mappings between goals
- Value:
  - Objects: efficiency, safety, precision, etc.
  - Morphisms: relations between values

CORESENSE

The SYSSELF Category

$$
\begin{tikzcd}
\mathit{Component} \arrow[r, "\mathit{enables}"] \arrow[d, "\mathit{realizes}"'] & \mathit{Goal} \arrow[d, "\mathit{aligned\ with}"] \\
\mathit{Capability} \arrow[r, "\mathit{contributes\ to}"] & \mathit{Value}
\end{tikzcd}
$$

Intelligent Robotics Lab

CORESENSE

Value as Pushout

SYSSELF

- Identify designs that provide expected value
  - Value: Benefit at cost provided to stakeholders
  - Pushout: Best approximation of an object satisfying certain conditions

$$
\begin{tikzcd}
Component \arrow[r] \arrow[d] & Goal \arrow[d] \arrow[ddr, bend left, "max\_perf"] & \\
Capability \arrow[r] \arrow[drr, bend right, "min\_cost"'] & Value \arrow[dr, dashed, "u"] & \\
 & & Expected\_value
\end{tikzcd}
$$

Intelligent Robotics Lab

Adaptation: Yoneda lemma

- Adapt: apply a natural transformation (α) between two Realization

Categories which objects are “**the same**” from a certain perspective

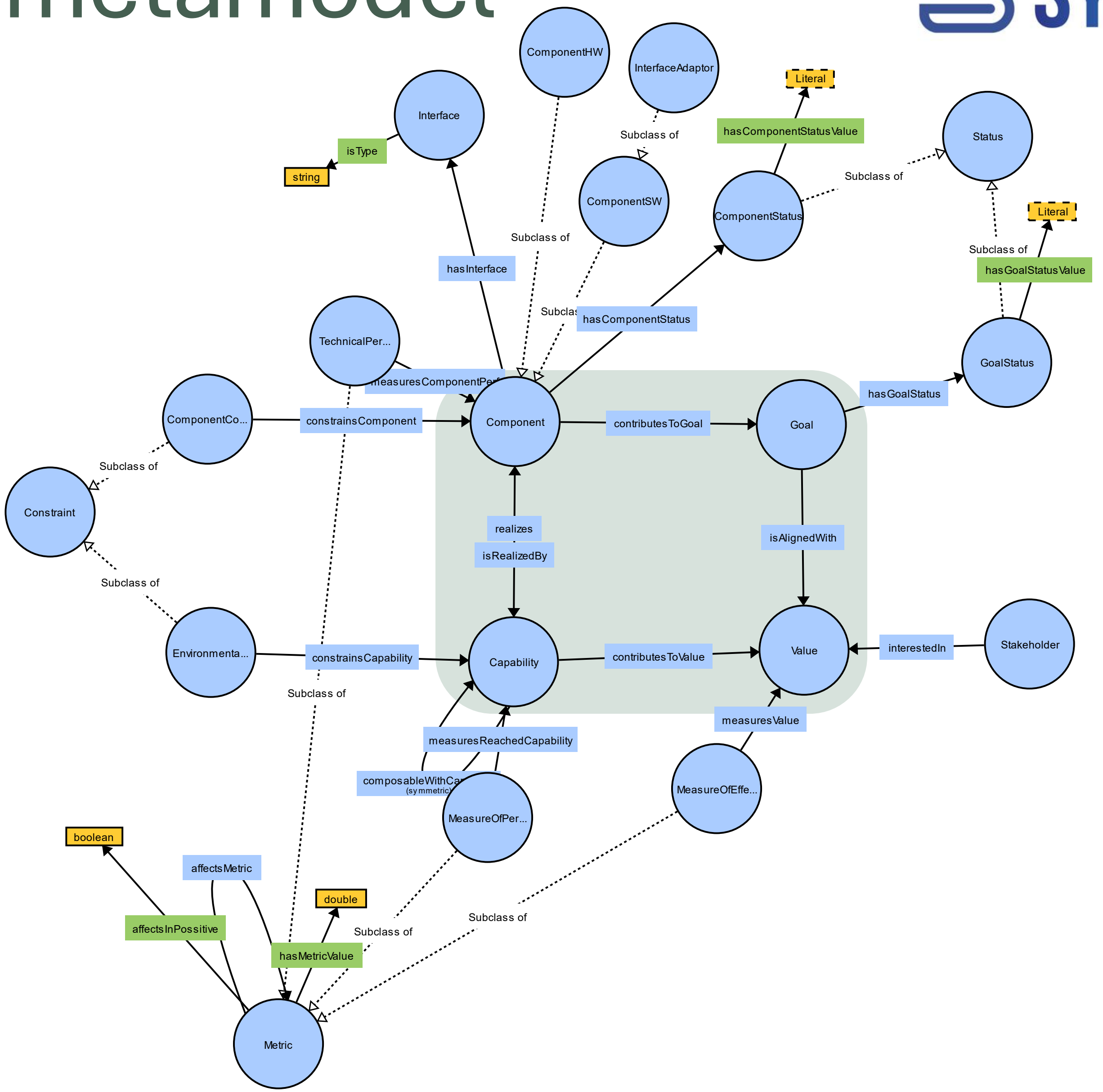
$$
\begin{tikzcd}
Realization \arrow[r, bend left=50, "\delta"{name=U}] \arrow[r, bend right=50, "\delta'"'{name=D}] & Realization' \arrow[from=U, to=D, Rightarrow, "\alpha"]
\end{tikzcd}
$$

Metrics

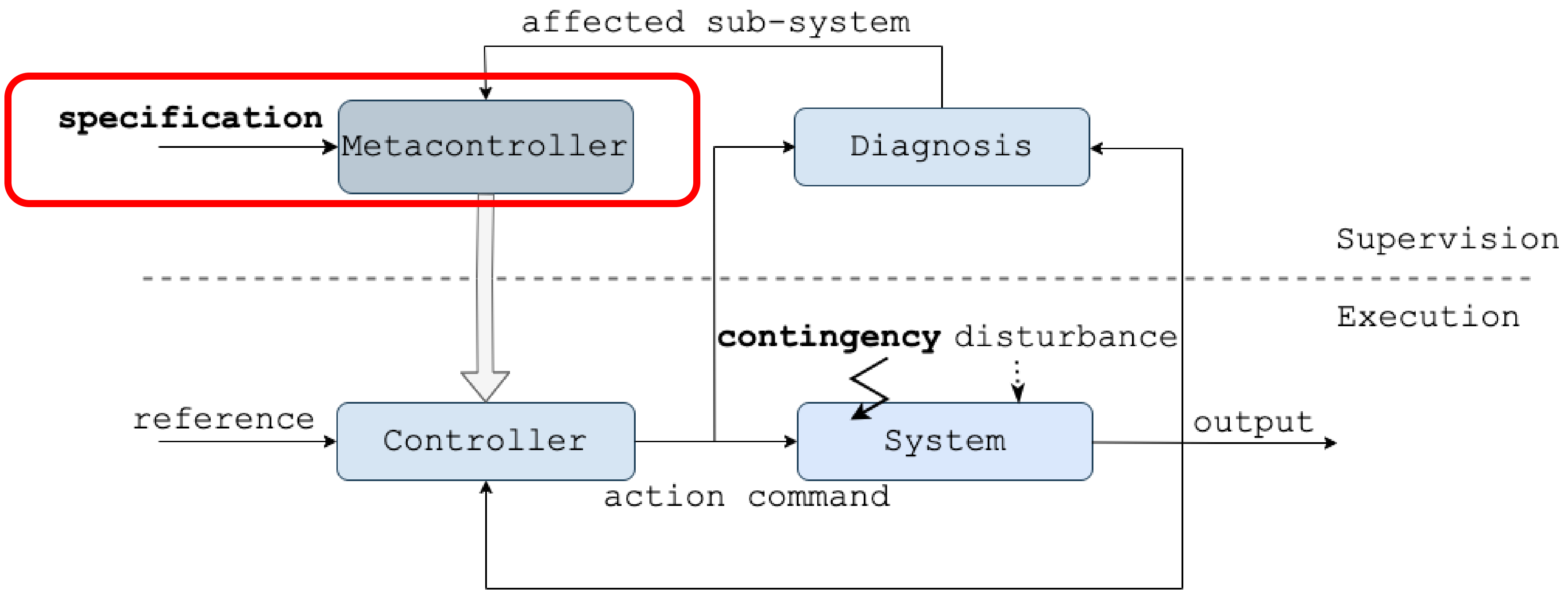
- Measure of Effectiveness (MOE) – Value
- Measure of Performance (MOP) – Capability
- Technical Performance Measure (TPM) – Component

Notify concerned agents about changes

Complete metamodel

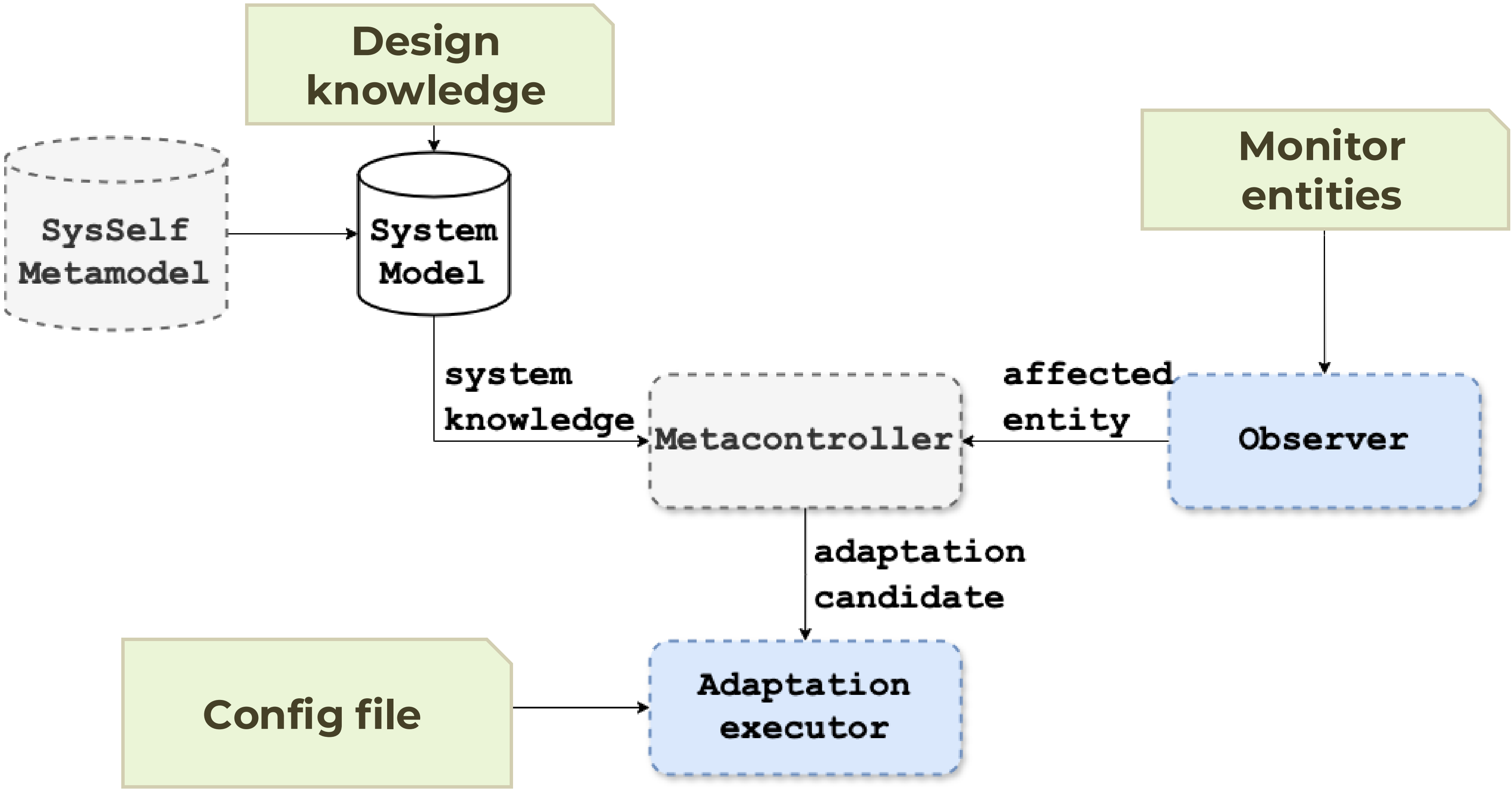


Metacontrol



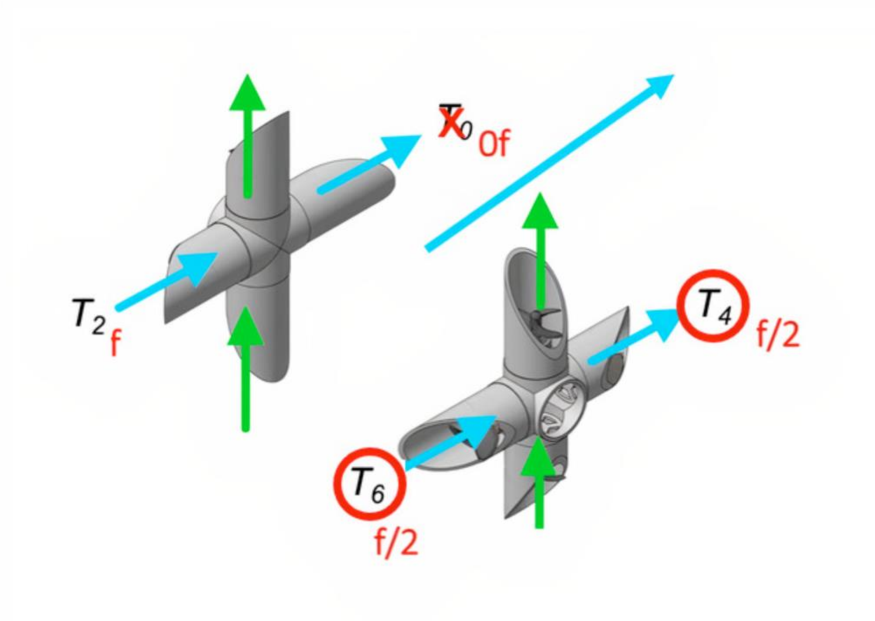
Architecture of a system using a metacontroller (An adaptive controller).

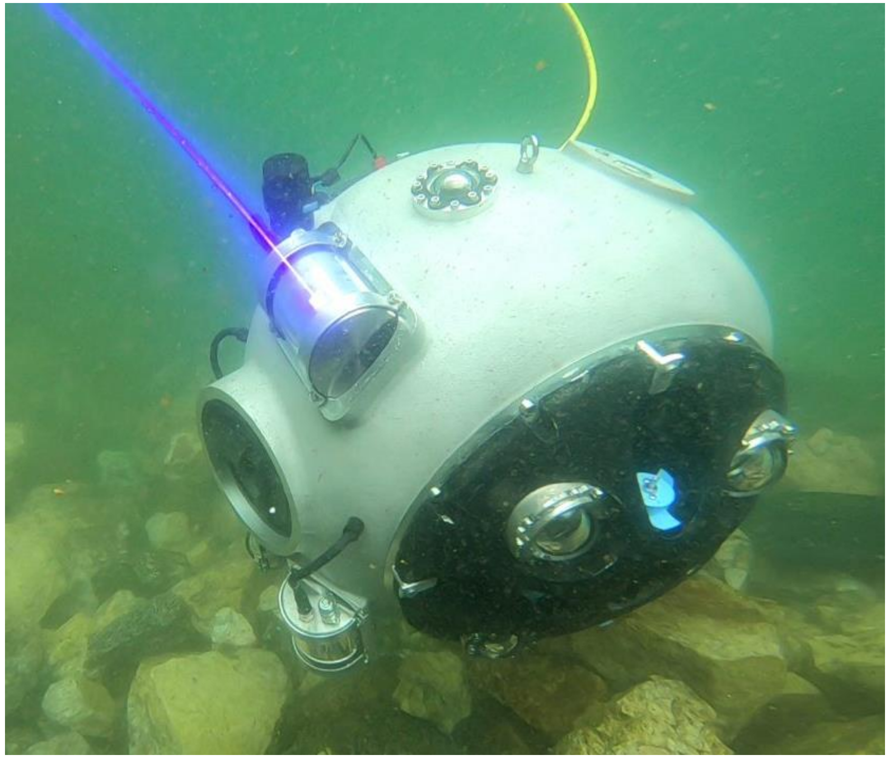
Usability



Design knowledge
SysSelf Metamodel
System Model
Monitor entities
system knowledge
Metacontroller
affected entity
Observer
adaptation candidate
Config file
Adaptation executor

Underwater mine robot

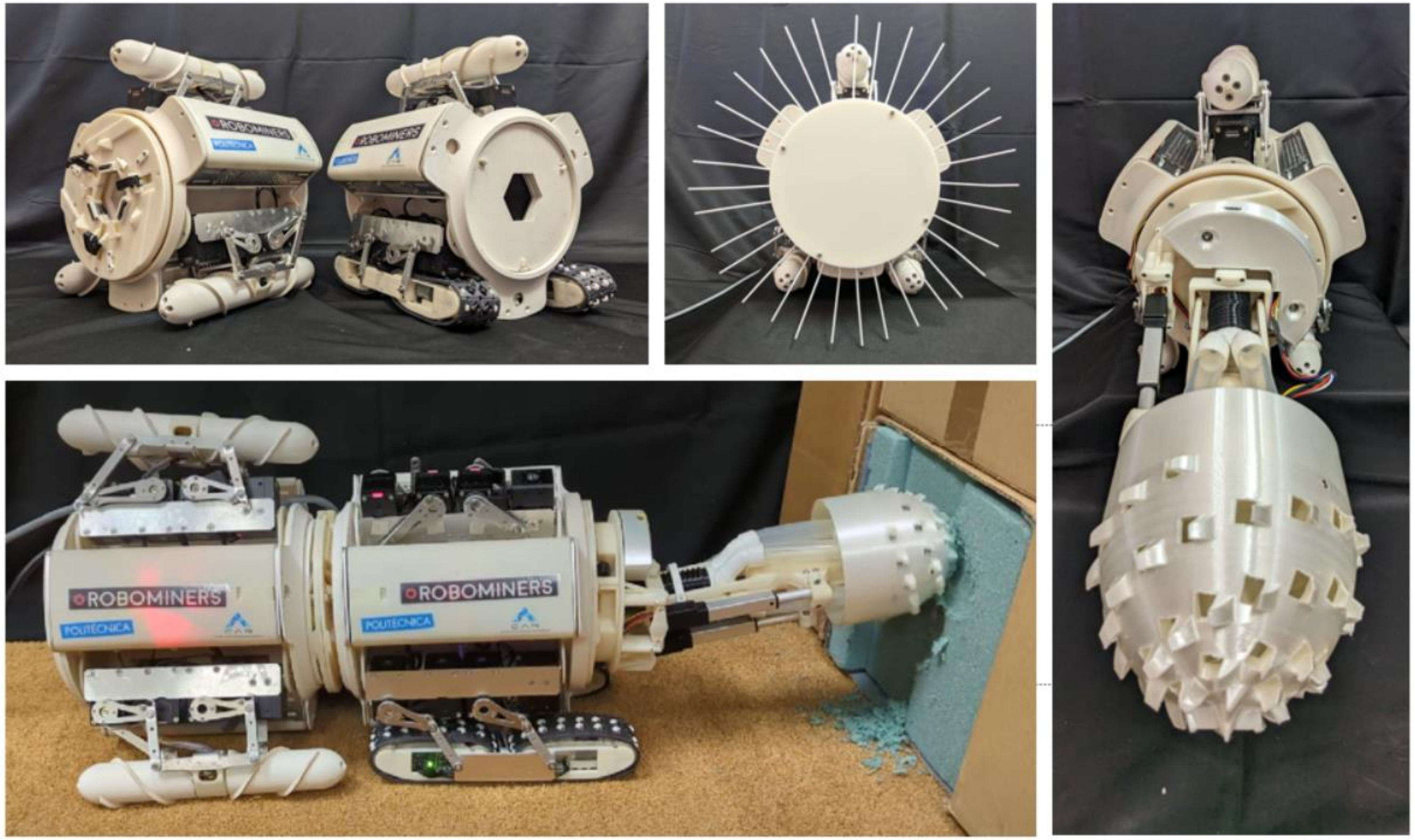




CORESENSE

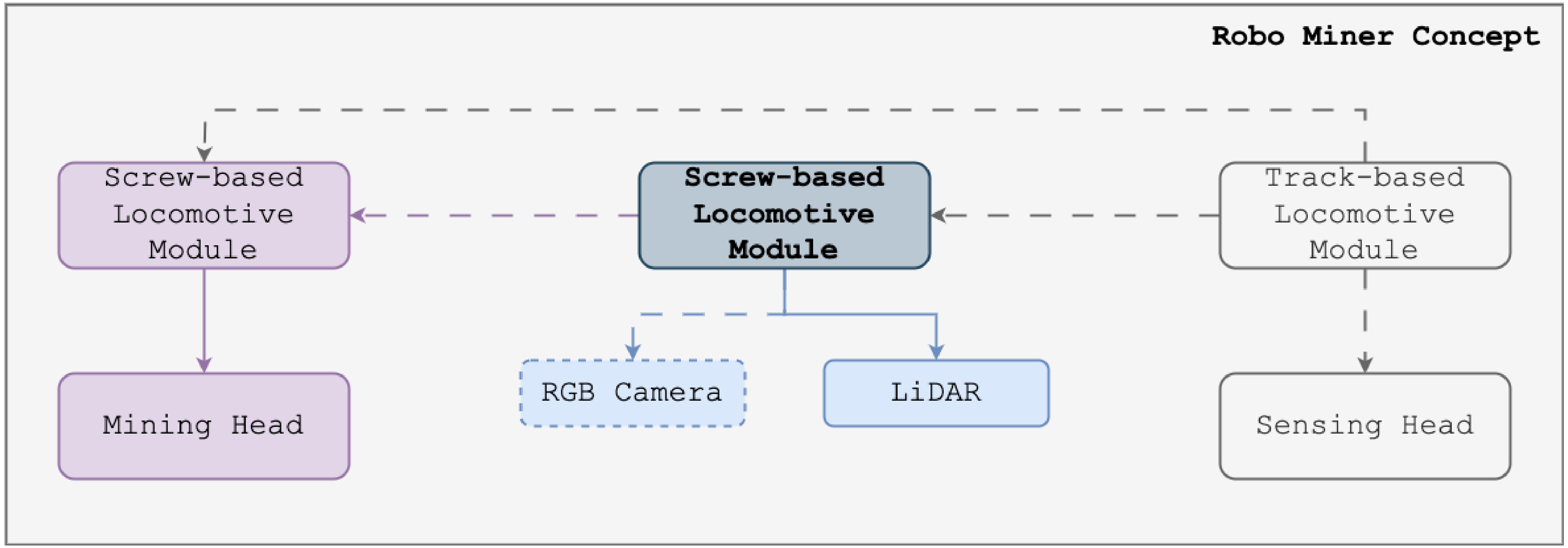
Applications: Modular Miner robot

ROBO MINERS



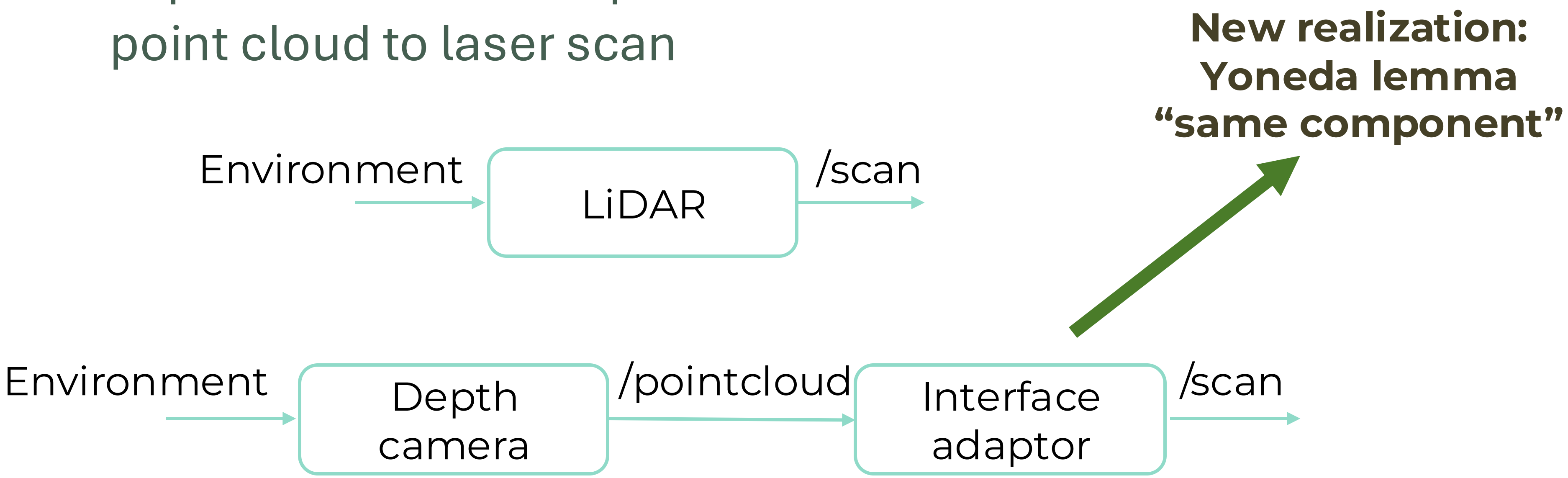
Intelligent Robotics Lab

Modular Miner robot

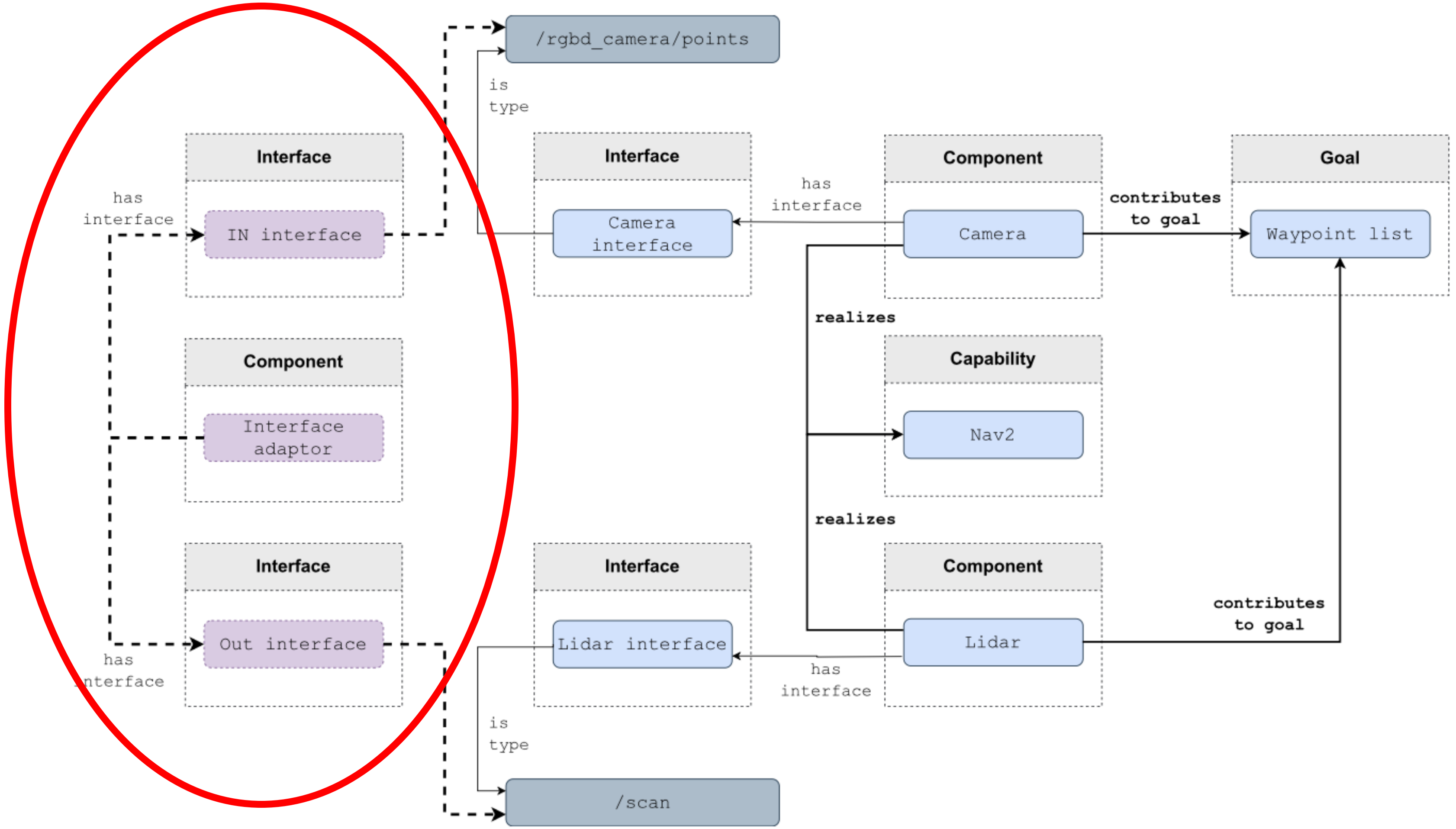


Failure in sensor

- LiDAR disconnected
  - Functional redundancy: depth camera
  - Requires interface adaption:
    point cloud to laser scan



Failure in sensor

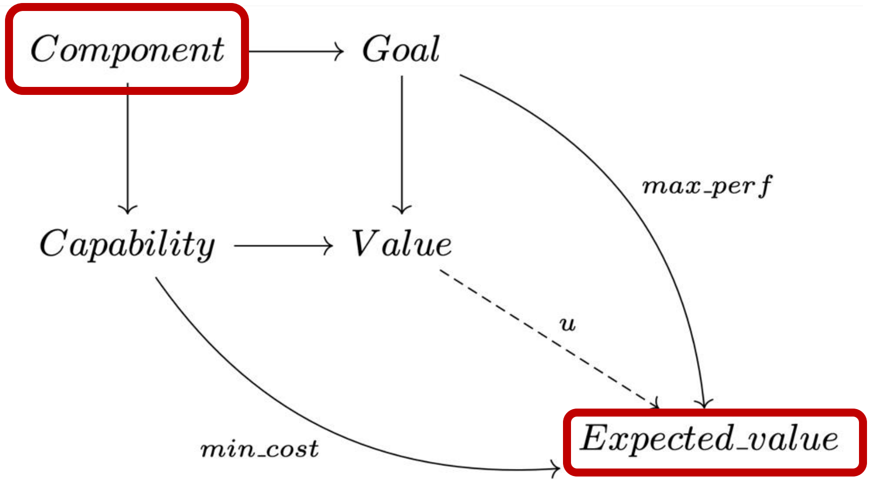


Intelligent Robotics Lab

Failure in sensor

Affected value:

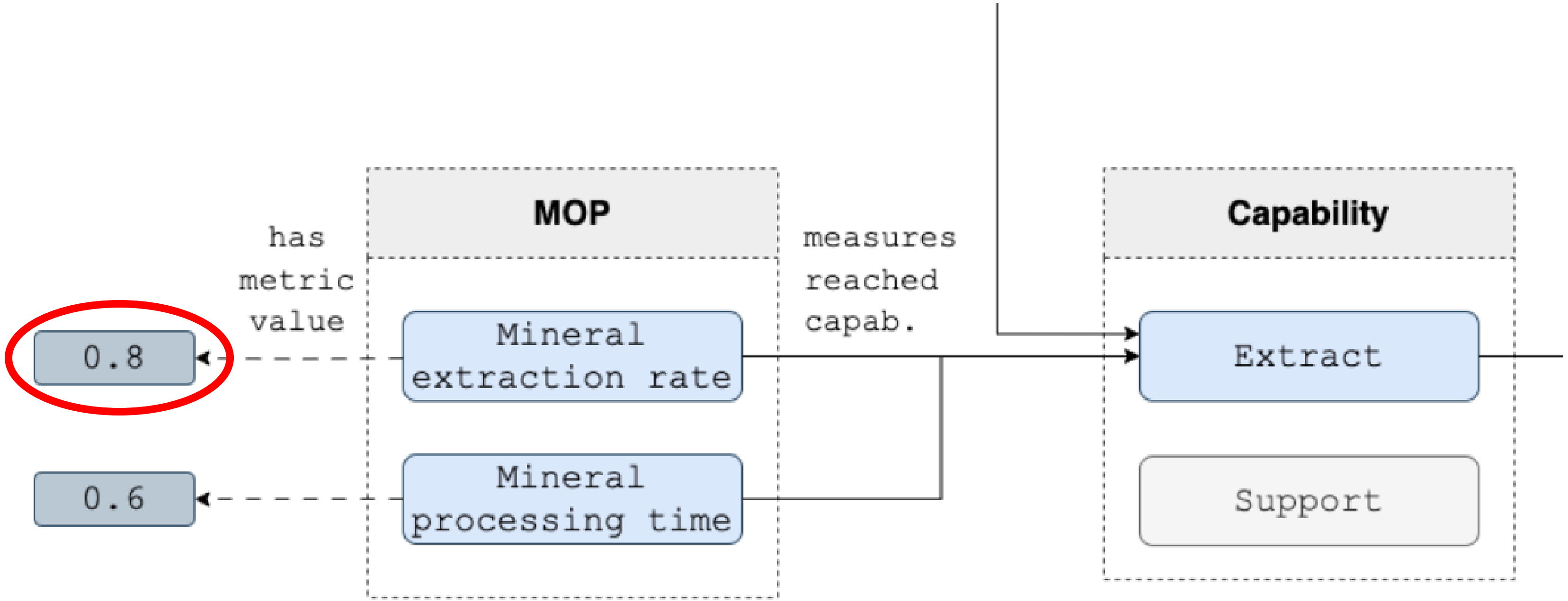
- Less point accuracy
- Less time efficiency
- Less energy consumption
- Task completed



Failure in sensor

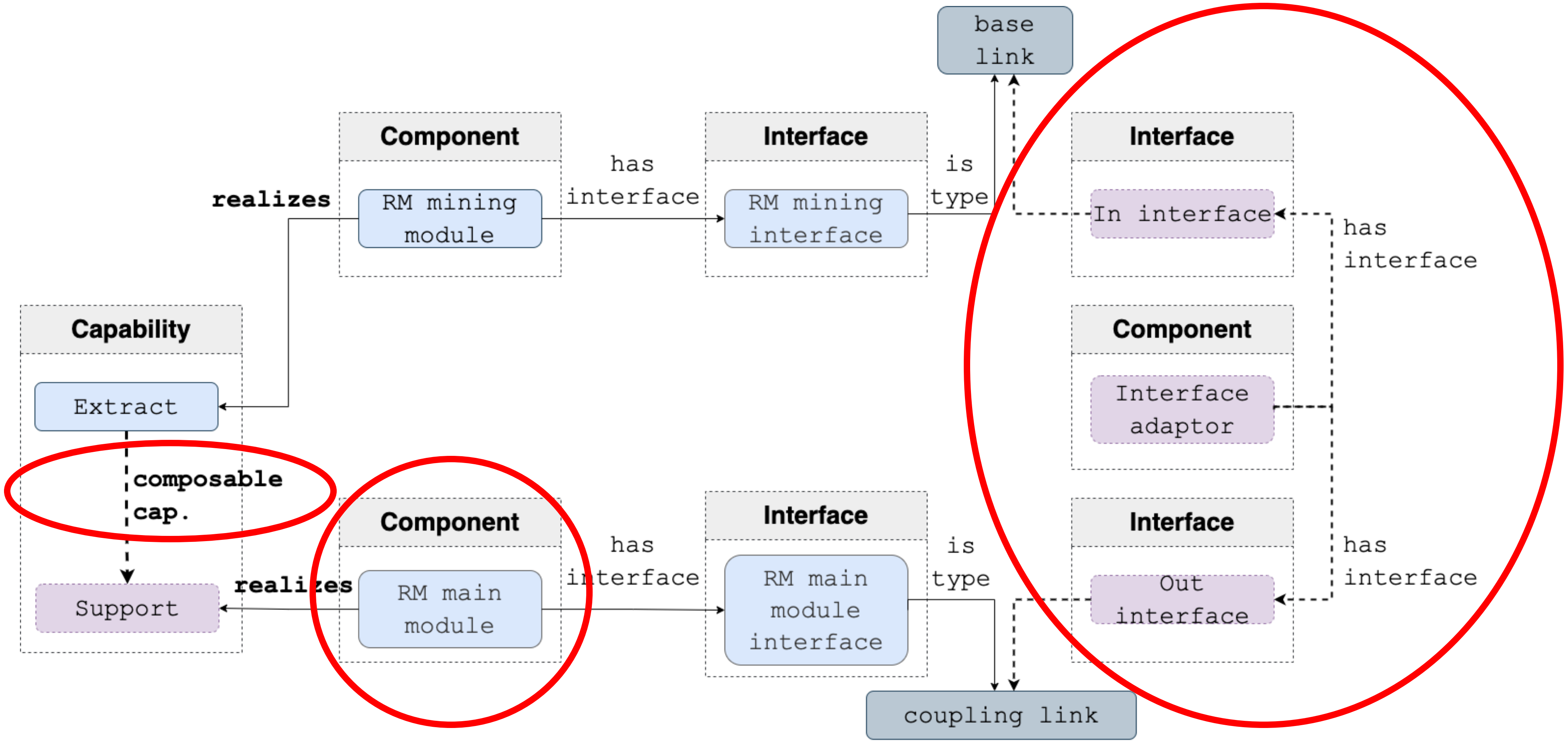
```
Initialization OK
Component lidar_status updated to value UNAVAILABLE
Component app_loc.camera AVAILABLE
REQUIRES app_loc.pointcloud_to_laserscan to be equivalent
Value value_robot_integrity DECREASED after adaption because
change in MOE mission_safety
Main stakeholder affected: robotic_worker

Value value_efficiency DECREASED after adaption because
change in MOE mission_duration
Main stakeholder affected: mine_worker

Value value_efficiency INCREASED after adaption because
change in MOE mission_energy_conssumption
Main stakeholder affected: mine_worker

New Configuration requested: [app_loc.camera,
app_loc.pointcloud_to_laserscan] of type ROS 2 NODE
 --------------------Reconfiguration successful--------------------
```

Decreased capability



Intelligent Robotics Lab

Decreased capability



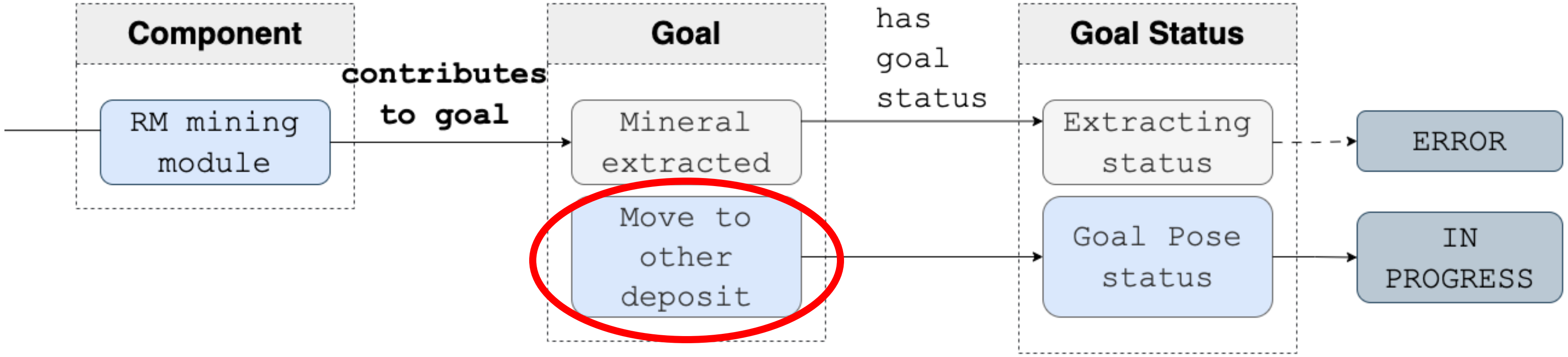
Decreased capability

```
Initialization OK
New WARN status received, checking metrics
Capability app_attach.capability_extract underachieved,
searching for alternatives
No alternative object found in Capability category
Searching for alternatives in other categories
Component app_attach.rm_main_module AVAILABLE
REQUIRES app_attach.interface_adaptor to be equivalent
Creating new morphism from relation in other category
---------------------Reasoner executed---------------------
```

Decreased capability

```
--------------------Reasoner executed--------------------
Value value_extraction INCREASED after adaption because
change in MOE mission_mineral_productivity
Main stakeholder affected: mine_exploiter

Value value_extraction INCREASED after adaption because
change in MOE mission_mineral_productivity
Main stakeholder affected: surface_operator

Value value_efficiency DECREASED after adaption because
change in MOE mission_duration
Main stakeholder affected: mine_operator

New Configuration requested: [app_attach.rm_main_module,
app_attach.interface_adaptor] of type ROS 2 NODE
--------------------Reconfiguration successful-----------------
Executing attach action
Action succeeded!
--------------------RESUMING TO MINING TASK------------------
```

Mission unreachable



Limitations

- Limited representation
  - Represent system evolution and risks
- Diagnosis
- Integration with other cognitive modules
- Extend evaluation:
  - Type of systems, metrics, engineering effort
- Steep learning curve
  - Model-2-model transformations

CORESENSE

Part 3:
What is next

Intelligent Robotics *Lab*

CORESENSE

A hybrid cognitive architecture

For deep understanding and awareness

The CoreSense project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 10107054

coresense.eu

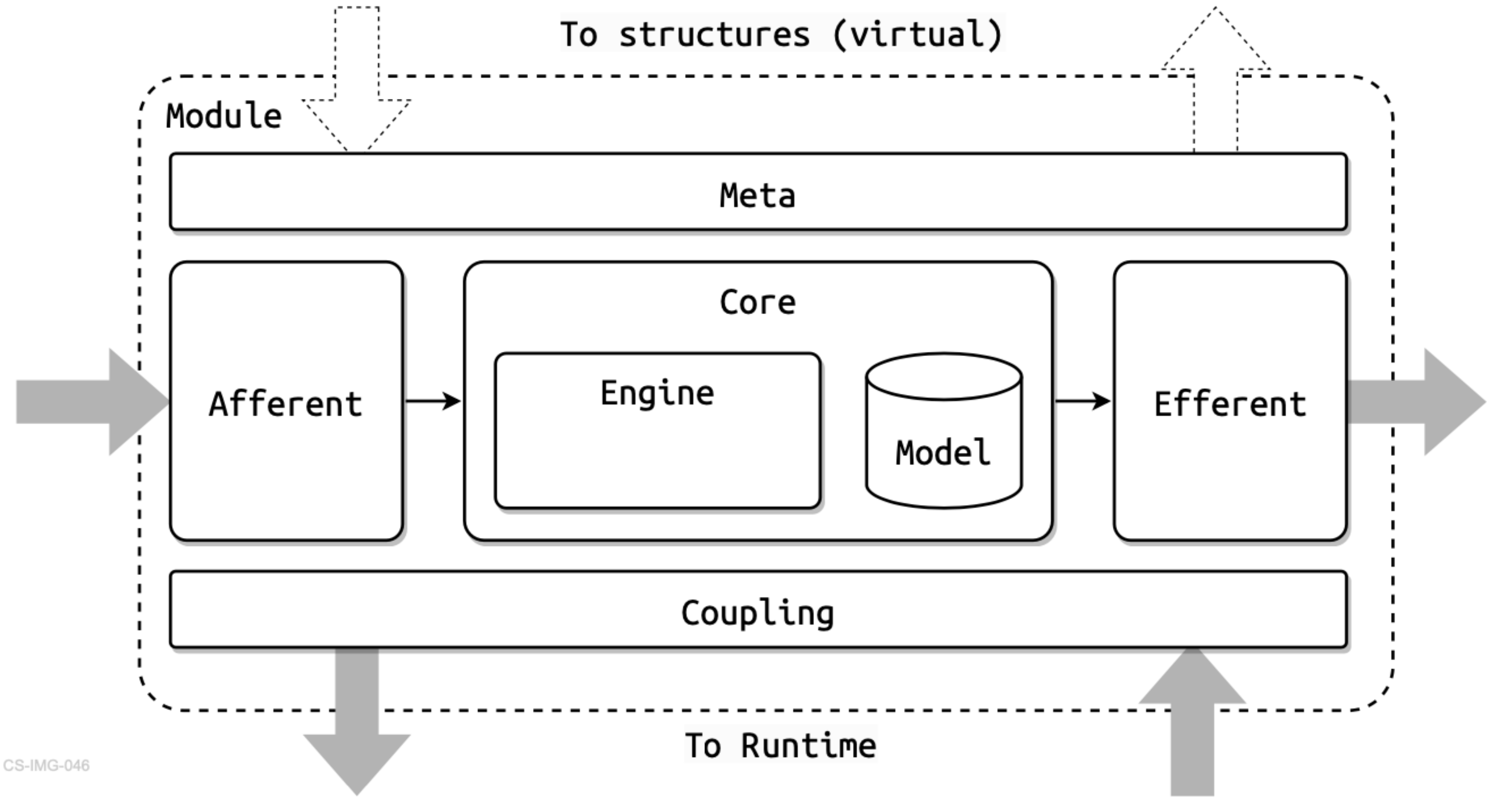
CoreSense: Problem

- Current limitations in intelligent robots:
  - Shallow understanding → rigid, predefined behaviours
  - Frequent failure in open or unexpected environments
- CoreSense aims to provide:
  - Deeper, dynamic, multi-actionable representations
  - Distributed cognitive capabilities
  - Increased adaptability, safety, and reliability

CoreSense: Approach

- Hybrid: Symbolic engineered models combined with data, geometrical, mathematical, etc.
- Exploit at runtime engineering model
- Value-oriented: prioritizes delivering the expected value to the end user
- Model-centric: used during the whole life-cycle

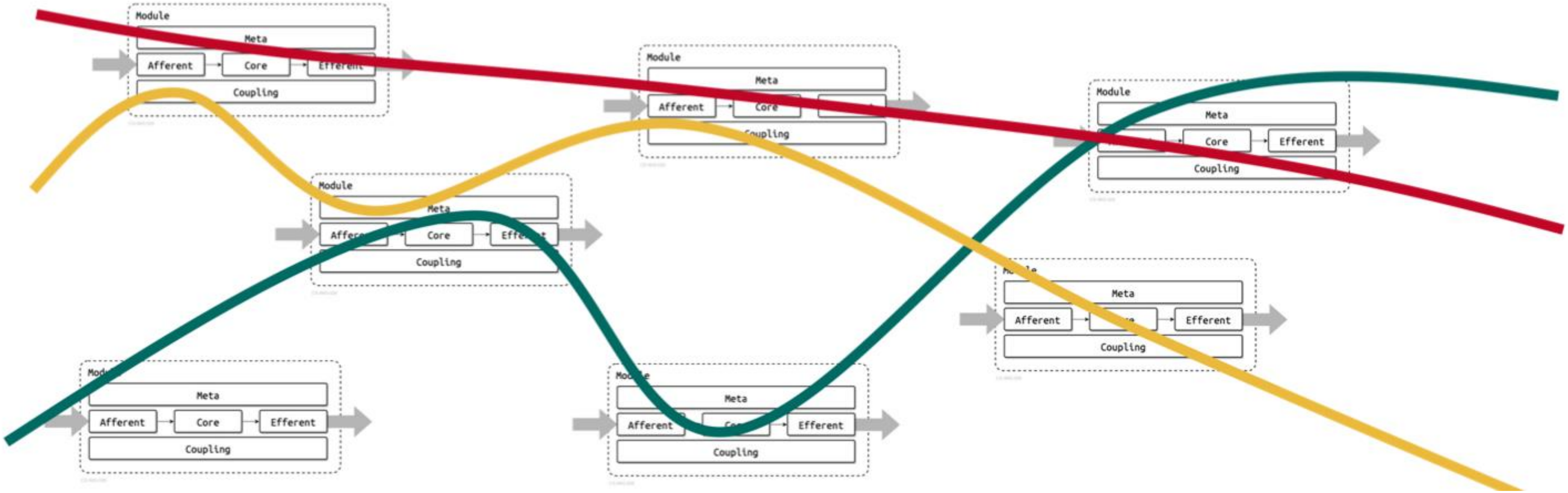
CoreSense: Aggregate of cognitive modules



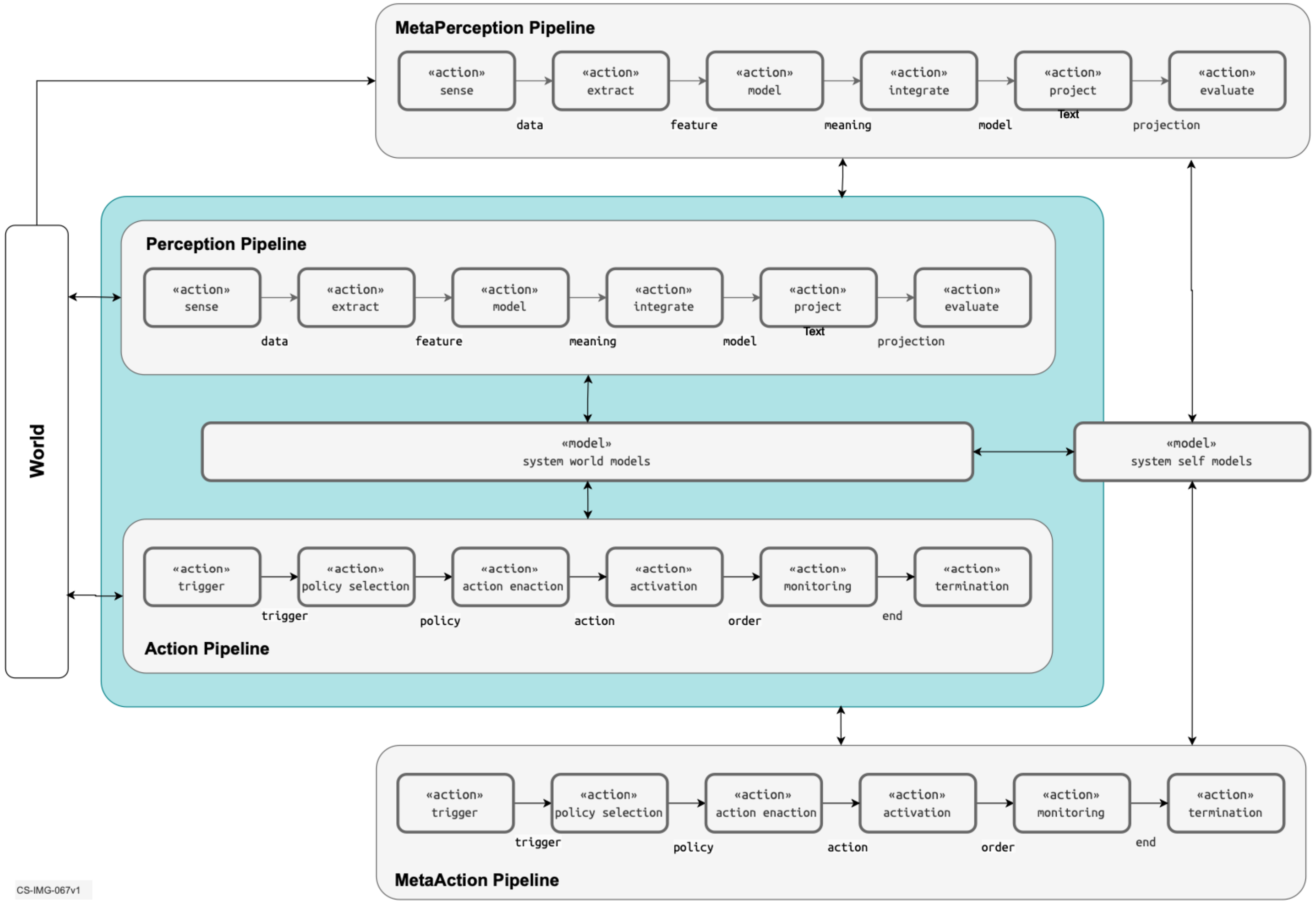
Intelligent Robotics Lab

CoreSense: Cognitive process

Distributed execution of cognitive functions



CoreSense: Fundamental essential



CORESENSE

Intelligent Robotics Lab

CoreSense: Reusability and applicability

- Reference architecture for cognitive robotic systems
- Wide applicability: manufacturing mobile manipulators, inspection drones, social robots
- ROS 2 compatible
- Supports both greenfield and brownfield system integration
- Architectural framework: methods, patterns, and tools

Part 3.2:

What is next

Limitations and future work

Challenges and limitations of current Cognitive Architectures

- Most effort is invested in high-level abilities:
  - Action selection, memory, reasoning, metareasoning
- Incomplete support for full general cognitive capabilities
- Perception often downplayed:
  - Lack of deep conceptualization
  - Weak symbol grounding
  - Unrealistic attention mechanisms → Limited understanding
- In robotics, focus is on navigation and manipulation
  - Still lacks integration with perceptual understanding

Challenges and limitations of current Cognitive Architectures

- Lack of experimental validation
  - Few standardized benchmarks or metrics
- Memory handling issues
  - Memory often treated as discrete snapshots with timestamps, limiting temporal reasoning and life-long learning
- Scalability problems
  - Symbolic knowledge bases struggle with real-time demands

AI trends and Cognitive Architectures

DL capable of solving AI?

Google DeepMind, Facebook AI research, etc. are working in:

- Solving important issues in AI: natural language, perceptual processing, cognitive abilities in limited domains
- No unified model of intelligence
- Approach: AI too complex to be built at once, focus on specific tasks

Future Work

- Advanced memory models: Incorporate continuous, context-aware, and hierarchical memory representations
- Improved usability and integration tools: Create developer-friendly toolkits and middleware for seamless deployment
- Adaptive and Self-Aware Systems: Enhance metacognition and introspection for robust autonomous behaviour under uncertainty
- Develop hybrid representations and reasoners at different level of abstraction
- System-wide capabilities

Part 3.3:

What is next

Conclusions

Take home ideas

- Cognitive architectures are reusable blueprints enabling robots to perceive, reason, learn, and act using knowledge.
- Classical systems (SOAR, ACT-R, LIDA) laid the groundwork but have limitations in real-world robotics
- Robotics frameworks (CRAM, SkiROS, etc.) are deployed in real robot and excel at specific tasks but face some limitations in scalability, adaptability, and usability challenges
- SysSelf approach advances robot self-awareness and metacognition but is not a full architecture, just a system-level module
- The CoreSense project pushes forward with hybrid architectures to overcome these issues

Conclusions

*Achieving deep, adaptive **understanding** in complex environments demands **overcoming** current **limits** in perception, knowledge integration, and memory management.*

*Hybrid cognitive architectures offer a promising path toward building **reliable** and **robust** autonomous robots.*

Cognitive Architectures for Robust and Reliable Robotics

Esther Aguado
esther.aguado@urjc.es

July 2025

CORESENSE

Intelligent Robotics Lab

Universidad Rey Juan Carlos

2nd ACM SIGSOFT Summer School for Software Engineering in Robotics

TU Delft

CORESENSE

References

Articles:

- Langley, P., Laird, J. E., & Rogers, S. (2009). Cognitive architectures: Research issues and challenges. Cognitive Systems Research, 10(2), 141-160. https://doi.org/10.1016/j.cogsys.2006.07.004
- Ingrand, F., & Ghallab, M. (2017). Deliberation for autonomous robots: A survey. Artificial Intelligence, 247, 10-44. https://doi.org/10.1016/j.artint.2014.11.003
- Vernon, D., Metta, G., & Sandini, G. (2007). A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. IEEE transactions on evolutionary computation, 11(2), 151-180. http://doi.org/10.1109/TEVC.2006.890274
- Antonio Lieto, Mehul Bhatt, Alessandro Oltramari, David Vernon, The role of cognitive architectures in general artificial intelligence, Cognitive Systems Research, Volume 48, 2018,Pages 1-3,ISSN 1389-0417. https://doi.org/10.1016/j.cogsys.2017.08.003
- Kotseruba, I., Tsotsos, J.K. 40 years of cognitive architectures: core cognitive abilities and practical applications. Artif Intell Rev 53, 17–94 (2020). https://doi.org/10.1007/s10462-018-9646-y

Intelligent Robotics Lab

References

Articles:

- Ritter FE, Tehranchi F, Oury JD. ACT-R: A cognitive architecture for modeling cognition. WIREs Cogn Sci. 2019; 10:e1488. https://doi.org/10.1002/wcs.1488
- Laird, J. E. (2022). Introduction to soar. arXiv preprint arXiv:2205.03854
- Stan Franklin, Tamas Madl, Steve Strain, Usef Faghihi, Daqi Dong, Sean Kugele, Javier Snaider, Pulin Agrawal, Sheng Chen, A LIDA cognitive model tutorial, Biologically Inspired Cognitive Architectures, Volume 16, 2016, Pages 105-130, ISSN 2212-683X, https://doi.org/10.1016/j.bica.2016.04.003
- Mayr, M., Rovida, F., & Krueger, V. (2023, October). Skiros2: A skill-based robot control platform for ros. In 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 6273-6280). IEEE. https://doi.org/10.1109/IROS55552.2023.10342216
- Michael Beetz, Gayane Kazhoyan, David Vernon, Robot manipulation in everyday activities with the CRAM 2.0 cognitive architecture and generalized action plans, Cognitive Systems Research, Volume 92, 2025, 101375, ISSN 1389-0417, https://doi.org/10.1016/j.cogsys.2025.101375

References

Books:

- Vernon, D. (2014). Artificial cognitive systems: A primer. MIT Press.
- Cangelosi, A., & Asada, M. (Eds.). (2022). Cognitive robotics. MIT Press. [Especially Ch 10: Cognitive Architectures]