An Analysis and Comparison of ACT-R and Soar

John Laird

University of Michigan

November 16, 2021

ACS 2021

#paper06-laird

Thanks to John Anderson, Dan Bothell, Nate Derbinsky, Steven Jones, Pat Langley, Christian Lebiere, Peter Lindes, Paul Rosenbloom, Bryan Sterns, and Andrea Stocco for comments on previous drafts, as well as the ACS reviewers.

Thanks to ONR and AFOSR for supporting this research.

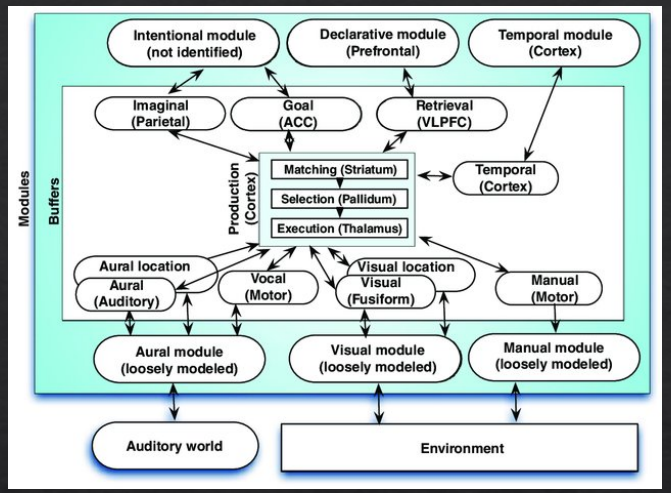
Comparing Cognitive Architectures

- ACT-R: Model human behavior + agents
- Soar: Complex cognitive agents + modeling

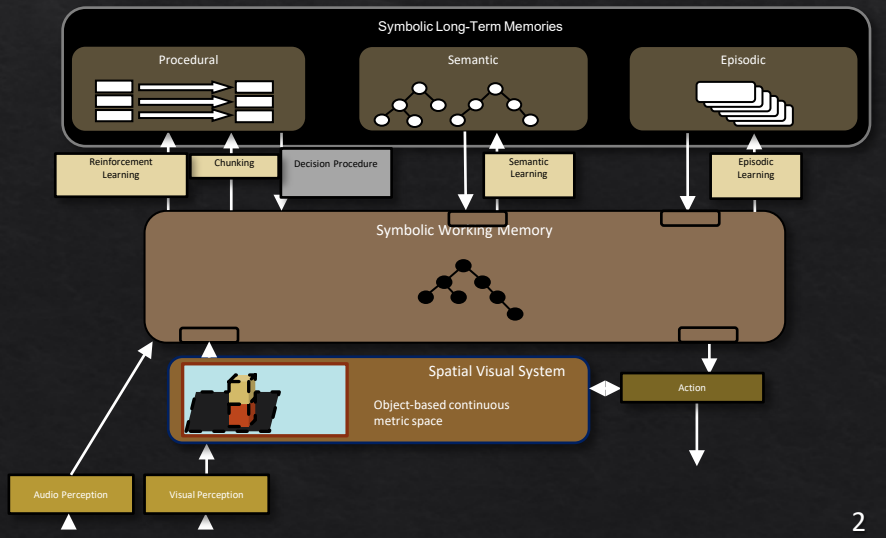
- Mature and general common model architectures
  - Freely available
  - Applied to 100's of tasks
  - Real-time performance
  - On-line incremental learning

- Deeper analysis than the Common Model.

ACT-R



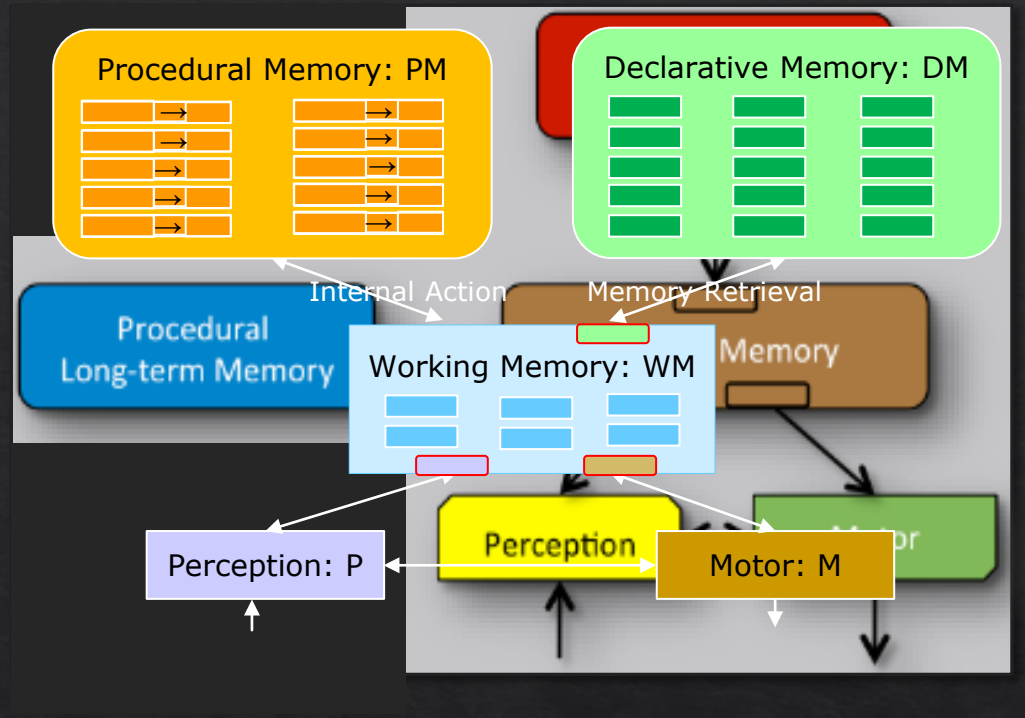
Soar



Common Model of Cognition

- Community consensus: an abstract specification of human-like cognitive architecture
  - Baseline for development and cumulative progress
  - Guide research on individual components
  - Testable theory for cognitive structures and functions
- *Not a prescription for all cognitive architectures*

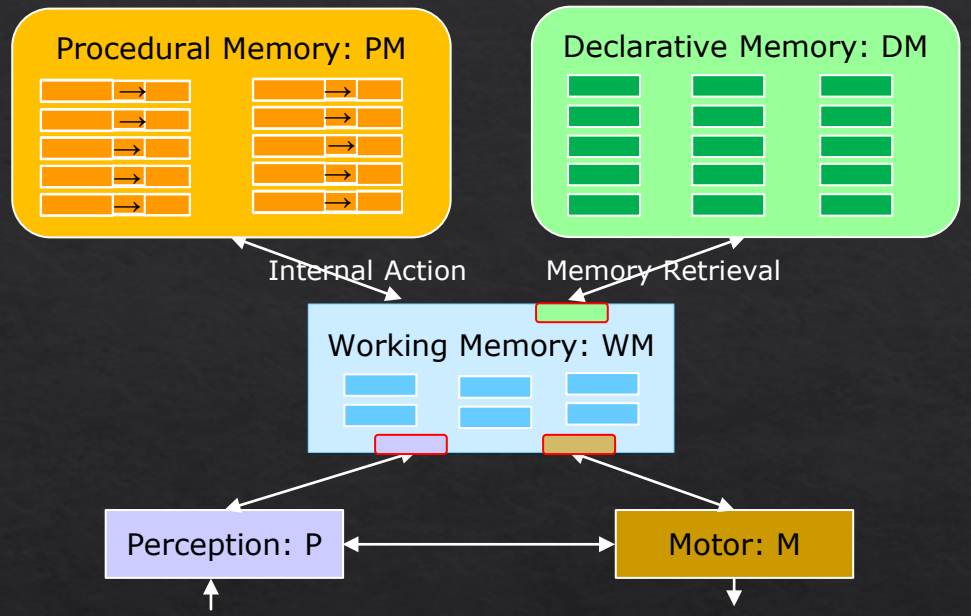
- A fixed set of communicating modules
  - Memories, perception, motor
  - Architectural learning



Laird, J. E., Lebiere, C. & Rosenbloom, P. S. (2017). A Standard Model for the Mind: Toward a Common Computational Framework across Artificial Intelligence, Cognitive Science, Neuroscience, and Robotics , *AI Magazine 38(4).*

Only Task-Independent Memory Modules

- No NL, planning, navigation, … modules
- No task-specific learning modules
- No executive control / metacognition / attentional modules

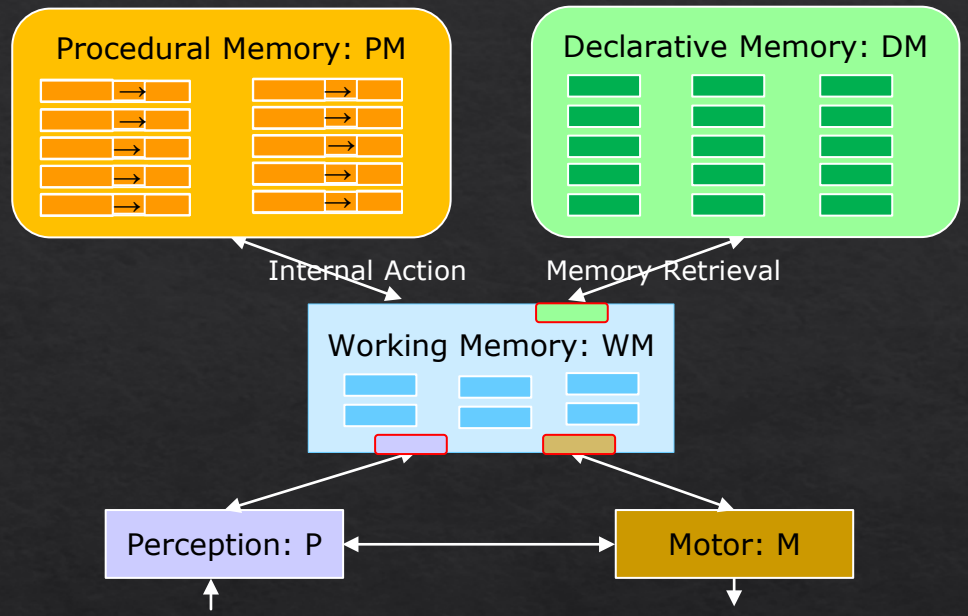


Outline: Commonalities and Differences

- Architecture Overview
  - Processing, Data, Metadata

- Working Memory (WM)
- Procedural Memory (PM)
- Declarative Memory (DM)

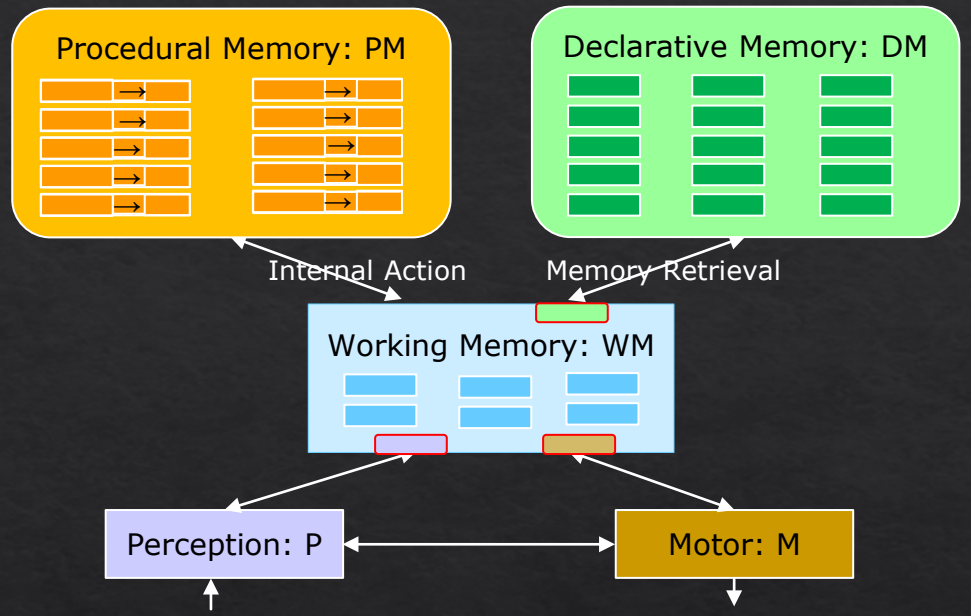
- Discussion: What did I learn?



Architecture Processing

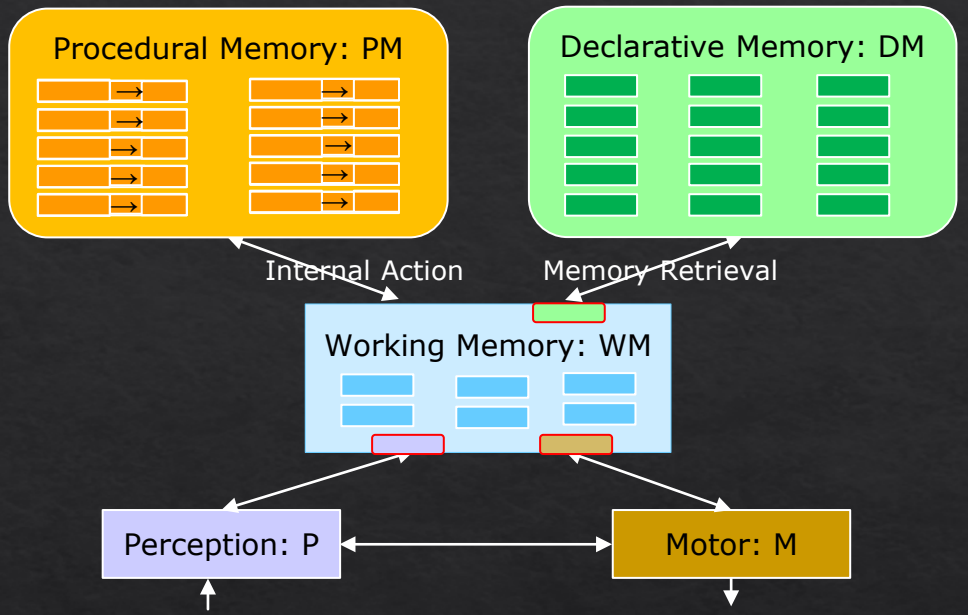
Basic Cycle: Selection and execution of an internal action from Procedural Memory to change Working Memory.

- Internal reasoning
- Declarative memory retrieval
- Motor action execution
- Perception modulation



Agent Data

- Encodes agent/task knowledge.
- Contents of working memory, declarative memory, procedural memory.
- Consist of *memory elements (symbolic).*
  - Independently created, modified, deleted, tested by other *agent data* and learning mechanisms.
- Architecture understands only the *form* of knowledge (some exceptions).

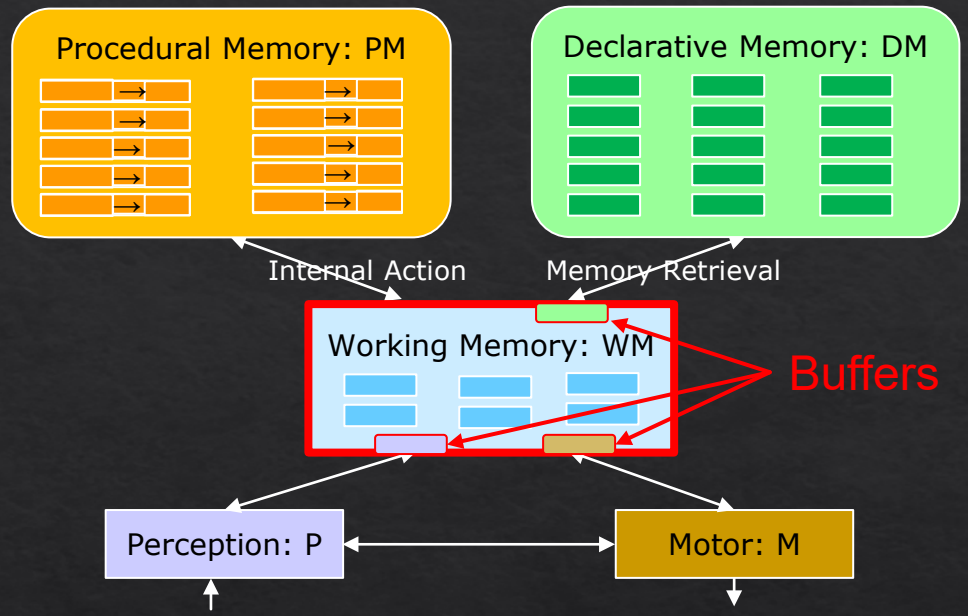


Types of Agent Data

1. **Internal agent data.**
   - Working memory, procedural memory, declarative memory.
   - *Unconstrained content.*

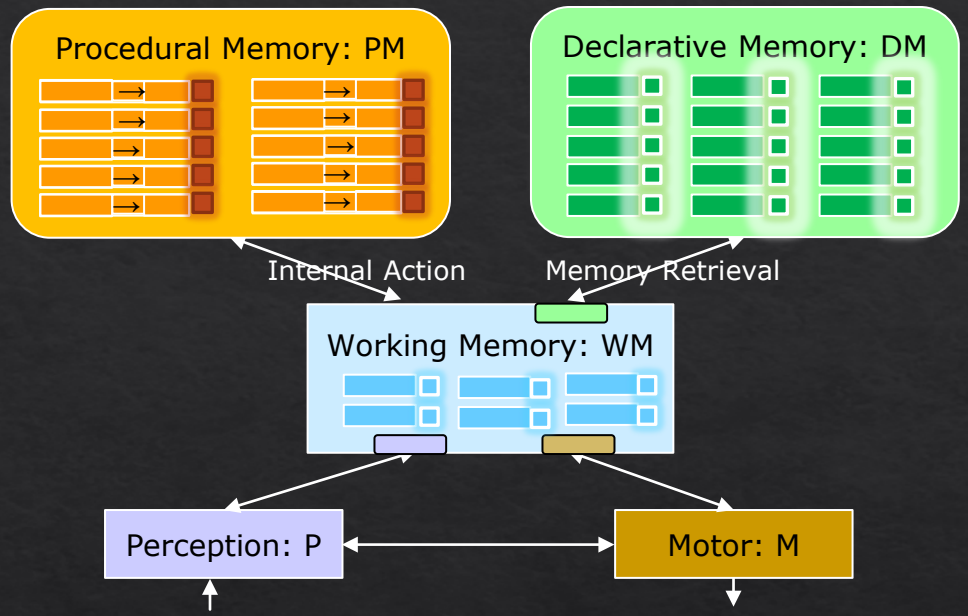
2. **Module commands.**
   - WM elements created by PM actions in WM *buffers*.
   - Sent to a module (P, DM, M) to initiate its processes.
   - *Innate/fixed set of symbols that define module commands.*

3. **Module status data.**
   - WM elements created by a module in its WM buffer.
   - Provide feedback on the module's processes: success/failure.
   - Meta-process data: data about architectural processing.
     - Signal to initiate metareasoning
   - *Innate/fixed set of symbols.*



Agent Metadata

- Data *about* agent data elements
  - Associated with agent data
  - Not defined as data used in metareasoning
  - Numeric and relational
  - *Fixed semantics – architecturally defined*
- Examples:
  - Activation of long-term memory elements
  - Utility of procedural memory elements
  - Derivational data for working memory elements
- Created, updated, and tested by the architecture
  - *Not accessible or modifiable by agent data*
- Influences architectural processing of agent data
  - *Retrievals from PM and DM memory*
  - *Learning*
  - *Forgetting*

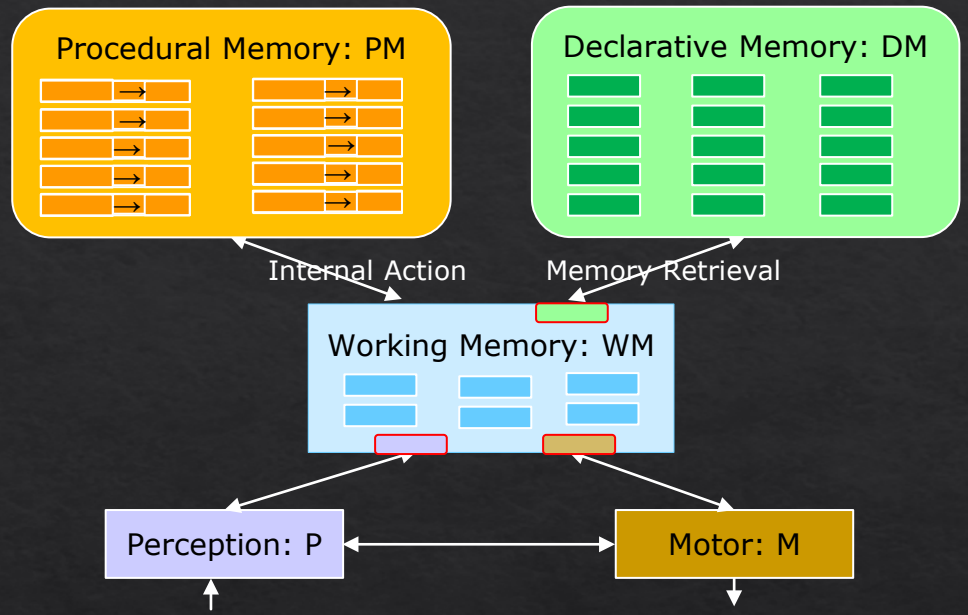


Outline: Commonalities and Differences

- ~~Architecture Overview~~
  - ~~Processing, Data, Metadata~~

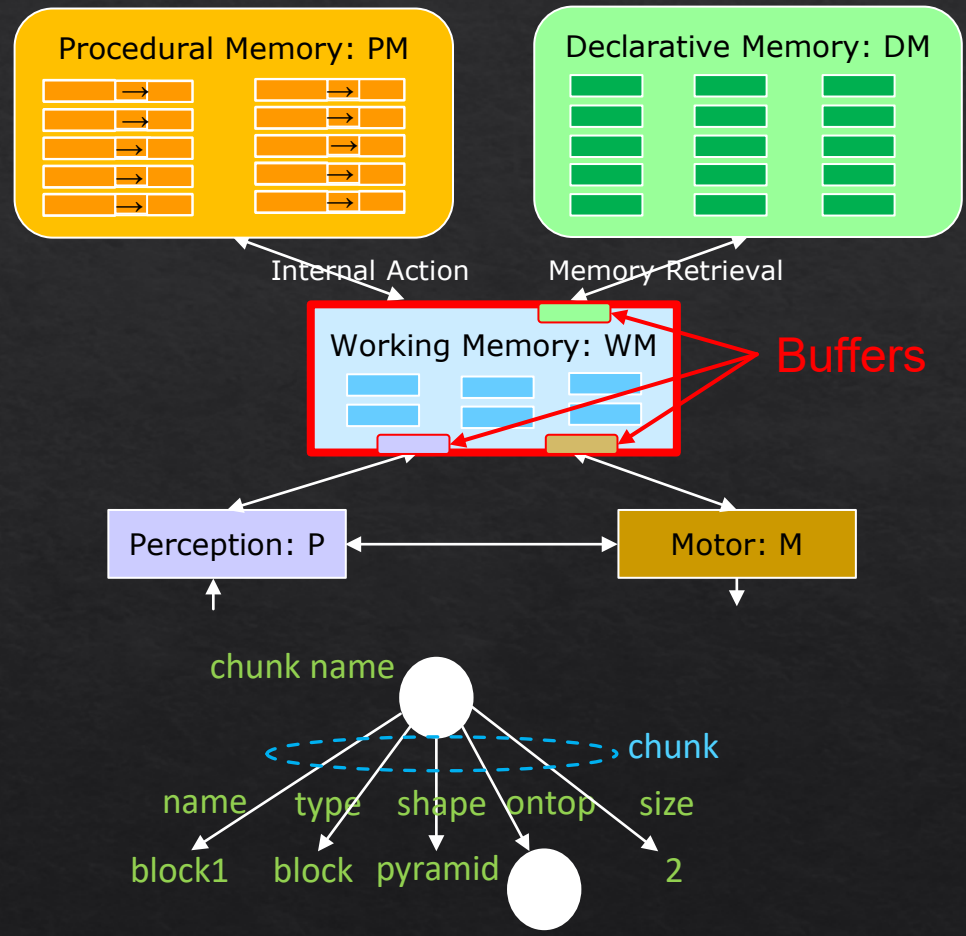
- Working Memory (WM)
- Procedural Memory (PM)
- Declarative Memory (DM)

- Discussion: What did I learn?



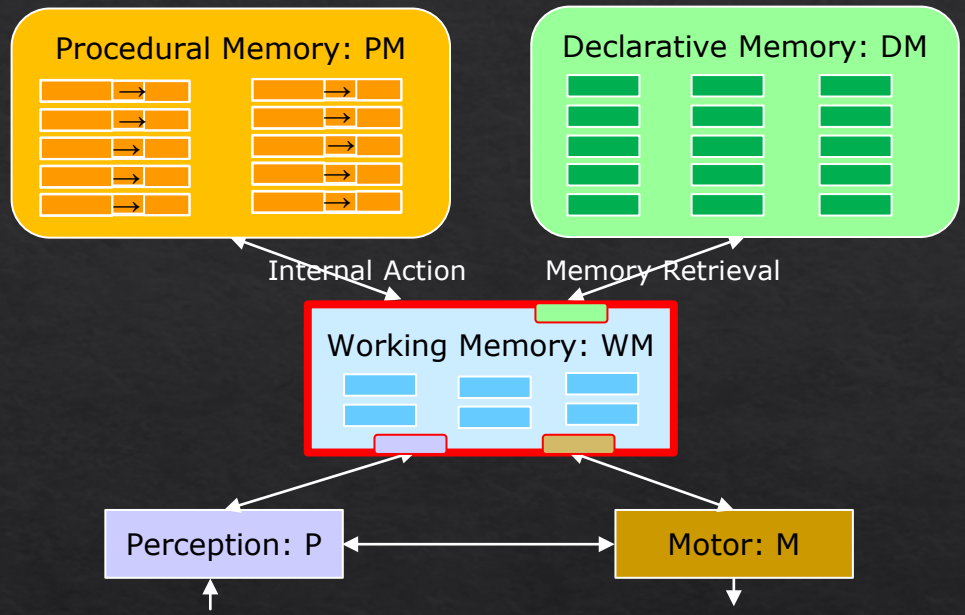
Working Memory Data Commonalities

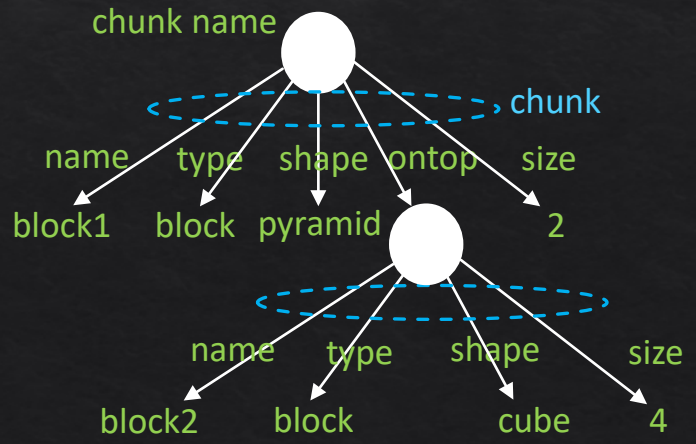
- Relational graph structures
  - *Element* is a triple: symbolic node, labeled edge, value.
  - *Declarative chunk*: Elements that share node
- Buffers to other modules: DM, PM, P, M
  - Chunks – data transfer to module
  - Module commands (read only – fixed semantics)
  - Module status (read only – fixed semantics)
    - Meta-process data



Working Memory Data Differences

- ACT-R
  - Fixed number of buffers – no other WM elements.
  - Exact number varies by agent/model
  - Single level chunks in buffers – no substructure.
- Soar
  - Unlimited graph in breadth and depth.
  - Rooted in *state* node.
  - Substates are created at impasse
    - Meta-process data for procedural memory.
    - Allows recursive “universal” subgoaling = metagoals.





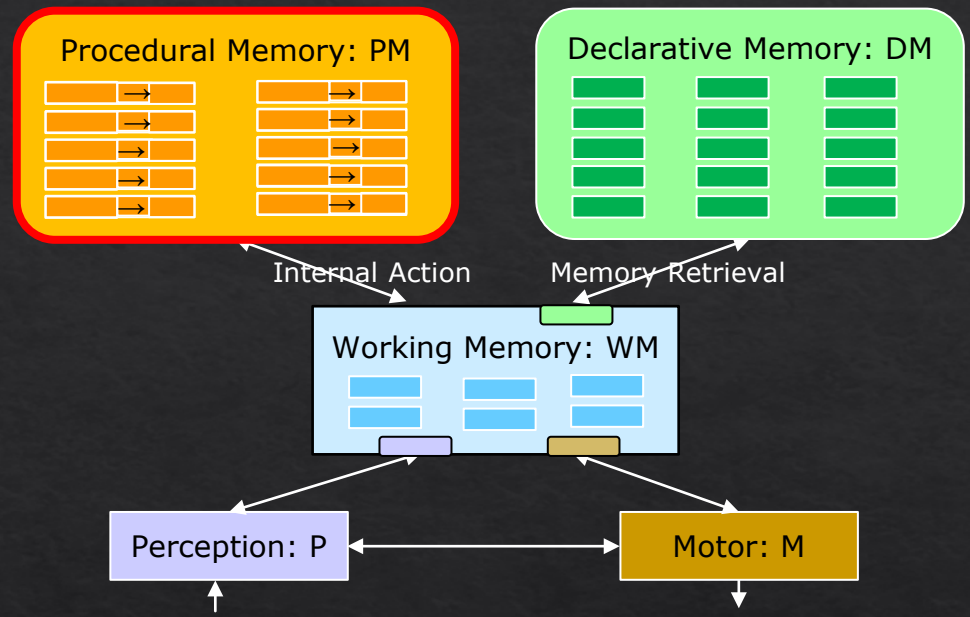
Working Memory Metadata

- Common
  - Source (relational):
    - Connection from WM chunk in DM chunk
    - Used for updating DM chunks and DM metadata
  - Derivation (relational):
    - Instantiation of procedural memory element used to create WM *elements*
    - Used in procedural learning

- ACT-R: above
- Soar: above +
  - Activation (numeric):
    - Base-level activation of element - recency and frequency of creation and access
    - Used in forgetting and for biasing DM retrievals
  - Highest substate (relational):
    - Used in returning results from substates and procedural learning

Procedural Memory Data

- Common:
  - A single PM element is selected on cognitive cycle
  - Rule-like elements with conditions and actions
    - Conditions test WM elements (not metadata)
    - Actions modify WM elements:
    - Updates metadata (PM utility, WM derivation, ...)
- ACT-R: above +
  - PM element = individual rule.
- Soar: above +
  - PM element = *operator* = collection of rules.
  - *Elaboration, proposal, selection, and application rules.*
  - Impasse if failed selection.

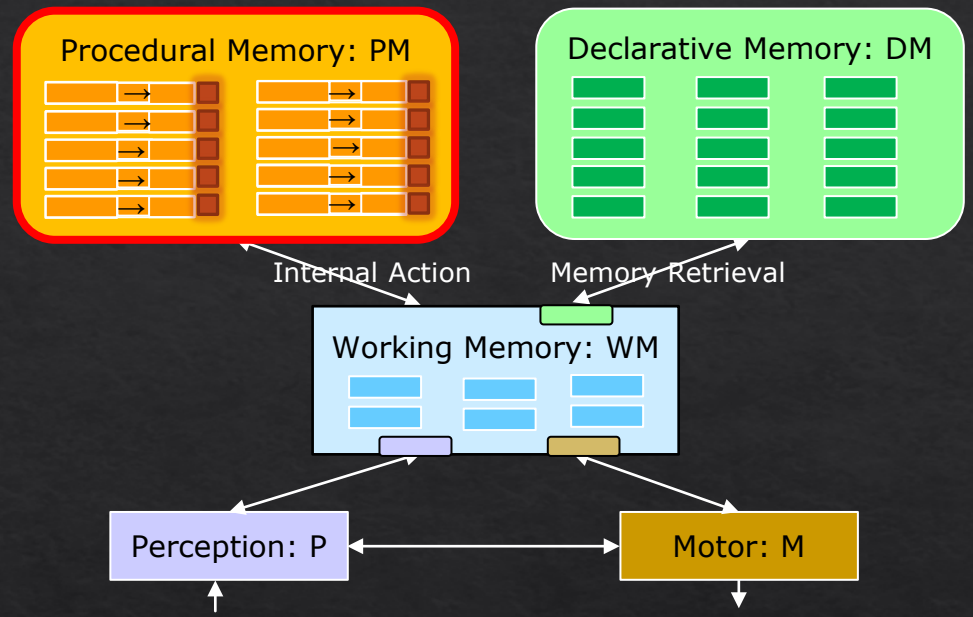


Procedural Memory Element: Rule vs. Operator

- ACT-R: Individual rule
  - Conditions match chunks in buffers
  - *Select rule based on utility and degree of match*
  - Selected rule actions change contents of buffers
- Soar: Operator
  - Elaboration rules: create WM structures tested in proposal and selection
  - Proposal rules: create proposed operator structures
  - Selection rules: create preferences with ratings of proposed operators
  - *Select operator based on preferences*
  - Selected operator apply rules change working memory data

Procedural Memory Metadata

- Common:
  - Utility associated with PM elements
  - Computed via temporal difference learning
- ACT-R: above
- Soar: above +
  - Utility associated with selection rules for an operator
  - Activation:
    - Recency and frequency of creation and access
    - Used in forgetting

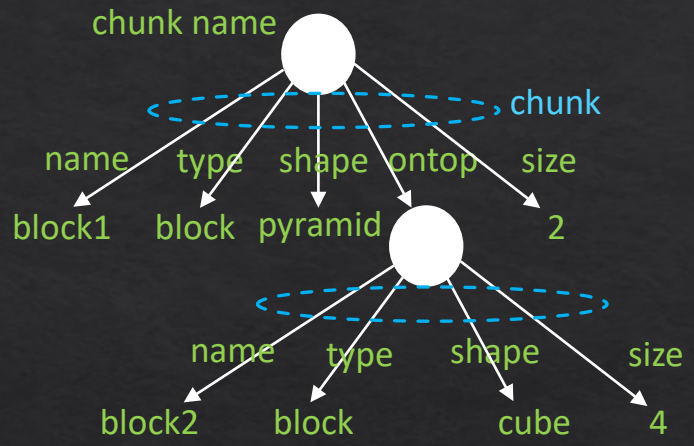


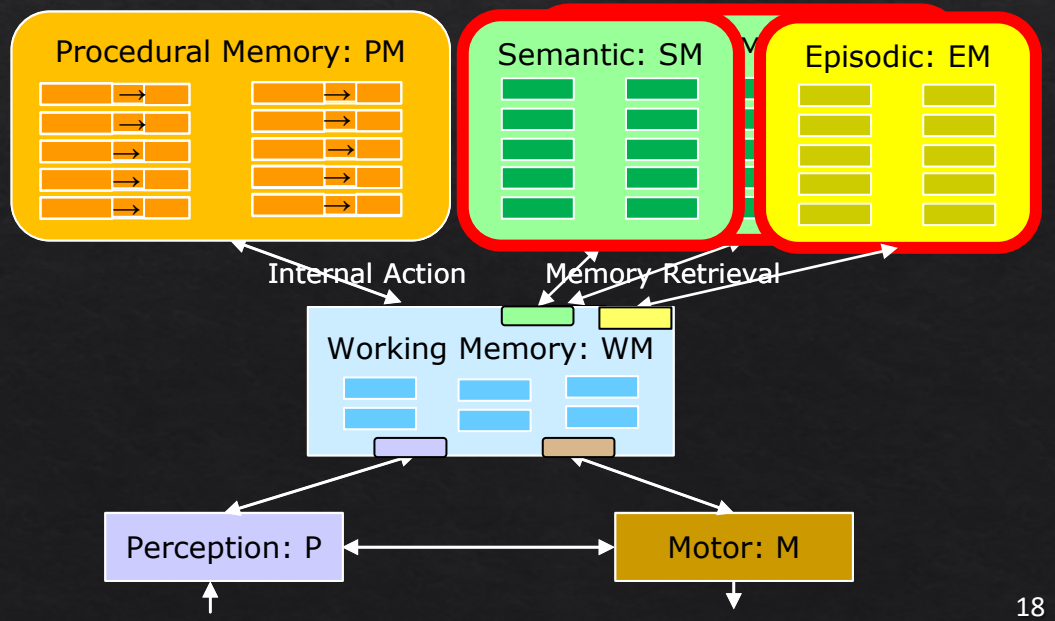
Procedural Memory Learning

- Common
  - RL is updated using PM elements utility metadata.
  - Procedural composition:
    - Use derivation metadata to create new PM elements that summarize processing
- ACT-R/Soar
  - Differences in details, but same functionality.

Declarative Memory Data

- Common:
  - Same relational graph structures as WM
- ACT-R: above
- Soar: above +
  - Semantic Memory = ACT-R Declarative Memory
  - Episodic Memory = Record of changes to WM





Declarative Memory Metadata

- Common
  - Base-level activation associated with every *chunk*
    - Computed from recency and frequency of access
    - Influences future retrievals from DM
- ACT-R: above +
  - Association strength between *pairs* of elements based on co-occurrence
    - Used in spreading activation
- Soar: above +
  - Base-level activation associated with every *element*
    - Used in spreading activation
  - Episodic memory has temporal and sequencing metadata for elements.

Declarative Memory Retrieval Differences

ACT-R: above +

- Select: Spontaneous Retrieval

Soar: above +

- Select: Direct Retrieval

Declarative Memory Learning

ACT-R

- Add chunk to DM when any buffer is cleared.
- Update DM chunk activation on chunk use and access.

Soar

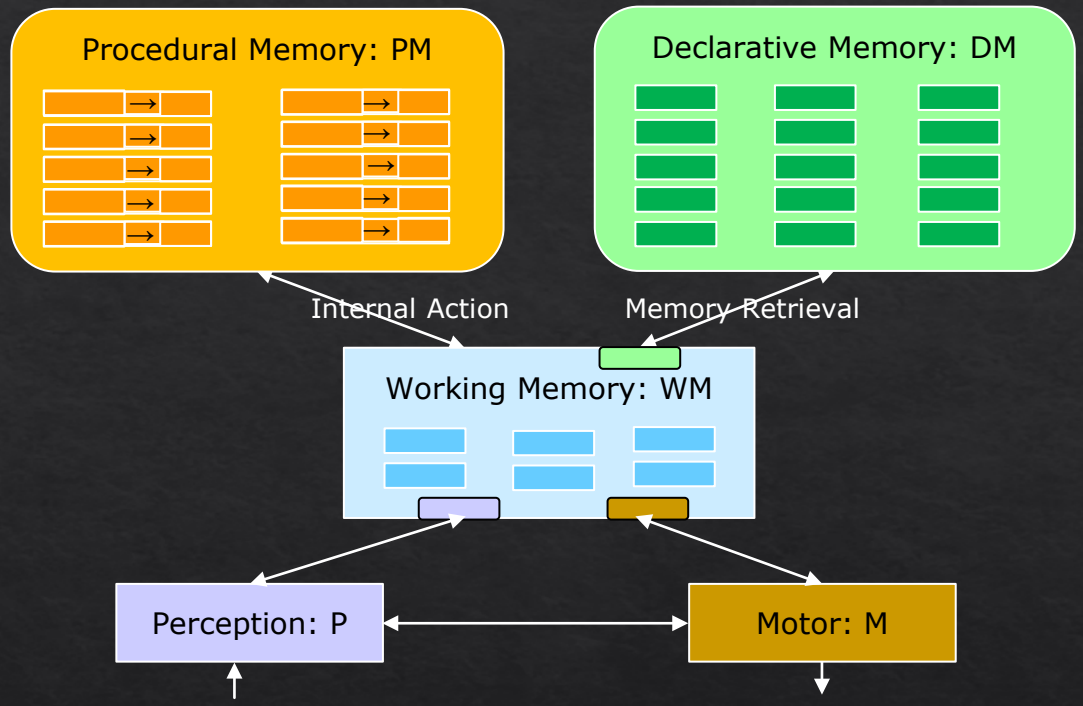
- Direct storage of WM chunk to Semantic Memory ☹
- Update DM chunk activation on chunk use and access.
- All changes to WM recorded in Episodic Memory.

Outline: Commonalities and Differences

- ~~Architecture Overview~~
  - ~~Processing, Data, Metadata~~

- ~~Working Memory (WM)~~
- ~~Procedural Memory (PM)~~
- ~~Declarative Memory (DM)~~

- Discussion: What did I learn?



Commonalities

1. Agent data represented as graph structures: elements, chunks, and rules
2. PM and DM retrievals biased by metadata
   - Change WM; update metadata; set module status
3. Modules interact through buffers in WM
   - Provide meta-process data
4. Agent Metadata
   - Numeric and relational
   - Updated in parallel with retrievals from PM and DM.
   - Not readable or modifiable.
   - Activation used to focus on important memory elements for selection and learning.
5. Learning is side-effect of PM and DM activity

Most Important Differences

ACT-R

- WM: Fixed set of buffers
- PM elements: single rules
- PM retrieval is all or none
  - And no indication of failure
- Single declarative memory

Soar

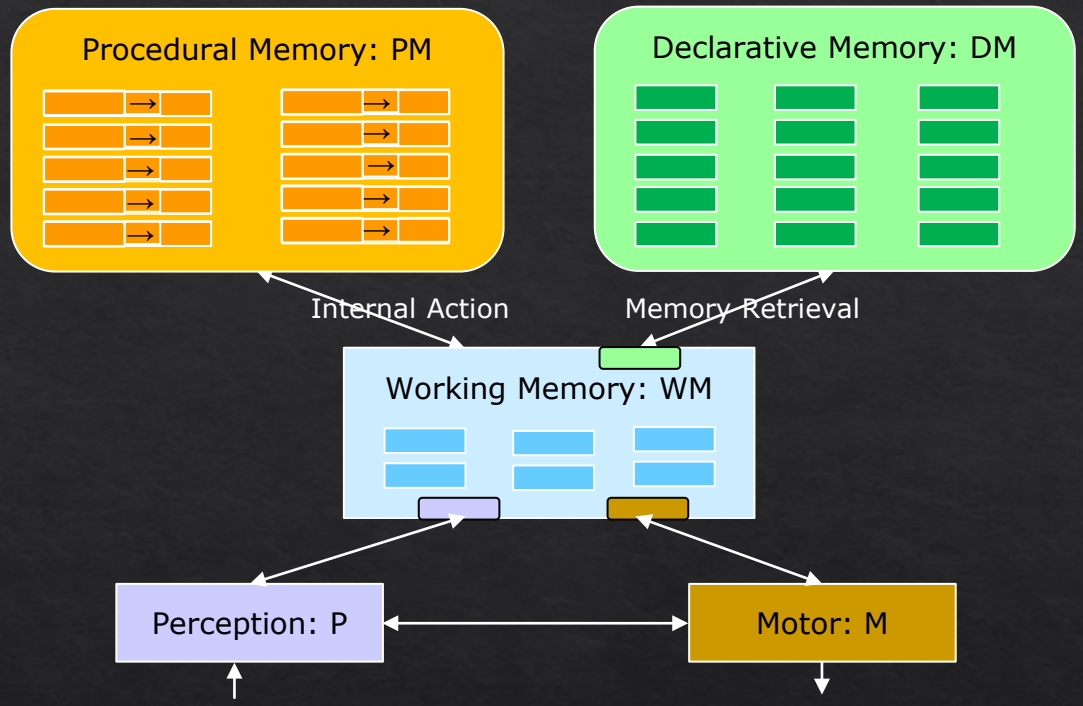
- WM: Unconstrained graph structure
  - Multiple states
- PM elements: operators
  - Multiple rules
- PM retrieval can fail in multiple ways
  - Impasses with substates
- Semantic and episodic memory

3 Types of Information

- Agent Data:
  - Created by perception, retrievals from procedural and declarative memory, and architectural learning mechanisms
  - Tested and modified by PM (except perception)
- Agent Metadata:
  - Associated with Agent Data
  - Created by architectural processing as a side effect of agent reasoning.
  - Influences architectural processing – retrievals and learning.
  - Does not “mix” with agent data – cannot be tested or modified by agent data.
- Module Status Data:
  - Meta-process data: created by module processing
  - Opening for metacognition
  - Tested by PM by cannot be modified

Only Task-Independent Memory Modules

- No NL, planning, navigation, … modules
- No task-specific learning modules
- No executive control / metacognition / attentional modules
  - Use same processing, but with access to meta-process data



Future

1. Analyze more architectures.
2. How represent metacognitive/metamemory appraisals?
   - Familiarity in a belief; feeling of knowing; feelings about ease of processing; judgments of remembering vs. knowing.
   - Status associated with retrievals from semantic and episodic memories?
   - Readable metadata?
3. Incorporate modality specific representations – mental imagery, …
4. Fold back into Common Model of Cognition