AI Architectures

or

State Requirements for Human-Like Action Selection

Joanna J. Bryson

Artificial models of natural Intelligence, University of Bath

Konrad Lorenz Institute for Evolution and Cognition Research

Why Action Selection?

Functionalist Assumption: All we care about is producing intelligent behaviour.

- Physical Symbol System Hypothesis (Newell & Simon 1963); Qualia, Chalmers “hard problem” (1995).
- Consciousness as epiphenomena (Churchland 1988, Brooks 1991).

We’ll build it if we need it.

Why Action Selection?

Functionalist Assumption: All we care about is producing intelligent behaviour.

- Physical Symbol System Hypothesis (Newell & Simon 1963); Qualia, Chalmers “hard problem” (1995).
- Consciousness as epiphenomena (Churchland 1988, Brooks & Stein 1993).

Science: We’ll build it to see if we need it.

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design

Outline

- Introduction
  - Intelligence, Cognition & Architecture
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design

Intelligence

- What matters is expressing the right behavior at the right time: action selection.
- Conventional AI planning searches for an action sequence, requires set of primitives.
- Learning searches for the right parameter values, requires primitives *and* parameters.
    - *parameter:* variable state.
    - Evolution and development *are* learning.

Combinatorics

- If . . .

  – an agent knows 100 actions (e.g. eat, drink, sleep, step, turn, lift, grasp, poke, flip...), and
  – it has a goal (e.g. go to Madagascar)

- Then . . .

  – Finding a one-step plan may take 100 acts.
  – A two-step plan may take 100^2 (10,000).
  – For unknown number of steps, may search forever, missing critical steps or sequence.

Intelligence & Design

- Combinatorics is the problem, search is the only solution.
- The task of intelligence is to focus search.
  - Called bias (learning) or constraint (planning).
  - Most `intelligent' behavior has no or little real-time search (non-cognitive).
- For artificial intelligence, most focus from design.

Cognition

Definition:

Cognition is on-line (real-time) search.

Consequence:

Cognition is bad.

Cognition

- Why is cognition / individual search bad?
  - Slow
  - Uncertain
- Unpopular in most species.
  - Plants

Cognition

- When is cognition useful?
  - Deeply dynamic environments -- change faster than learning or evolution can adapt.
  - Baldwin Effect -- speed up / facilitate slower learning processes (Baldwin 1896, Hinton & Nowlan 1987).

Architecture

- Where do you put the cognition?
- Really: How do you bias / constrain / focus cognition so that it works?

Architecture

- Where do you put the cognition?
- Really: How do you bias / constrain / focus cognition (learning, search) so it works?

Outline

- Introduction
  - Intelligence, Cognition & Architecture
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
  - SOAR/ACT-R, ANA (Maes Nets), Subsumption, BDI/PRS, CogAff, Brains
- Behavior Oriented Design

References

- Cyril Brom and Joanna J. Bryson, "Action Selection for Intelligent Systems", white paper for euCognition, 7 August 2006.
- Joanna J. Bryson, "Cross-Paradigm Analysis of Autonomous Agent Architecture", *Journal of Experimental and Theoretical Artificial Intelligence* **12**(2): 165-190, 2000.
- Joanna J. Bryson and Lynn Andrea Stein, "Architectures and Idioms: Making Progress in Agent Design", The Seventh International Workshop on Agent Theories, Architectures and Languages (ATAL), Boston, 2000.

“History as Evolution” Hypothesis

- If an architecture is around for a while, and it changes, the change was probably selected, adaptive.
  - This is particularly likely if the change goes against the stated theories of the architecture’s makers.

"History as Evolution" Hypothesis & Correlary

- If an architecture is around for a while, and it changes, the change was probably selected, adaptive.
- If similar features occur in a lot of architectures with different phylogenies, those features are probably adaptive.
  - If you want to make a contribution to a field, describe your best innovations in terms of well-known systems.

Productions

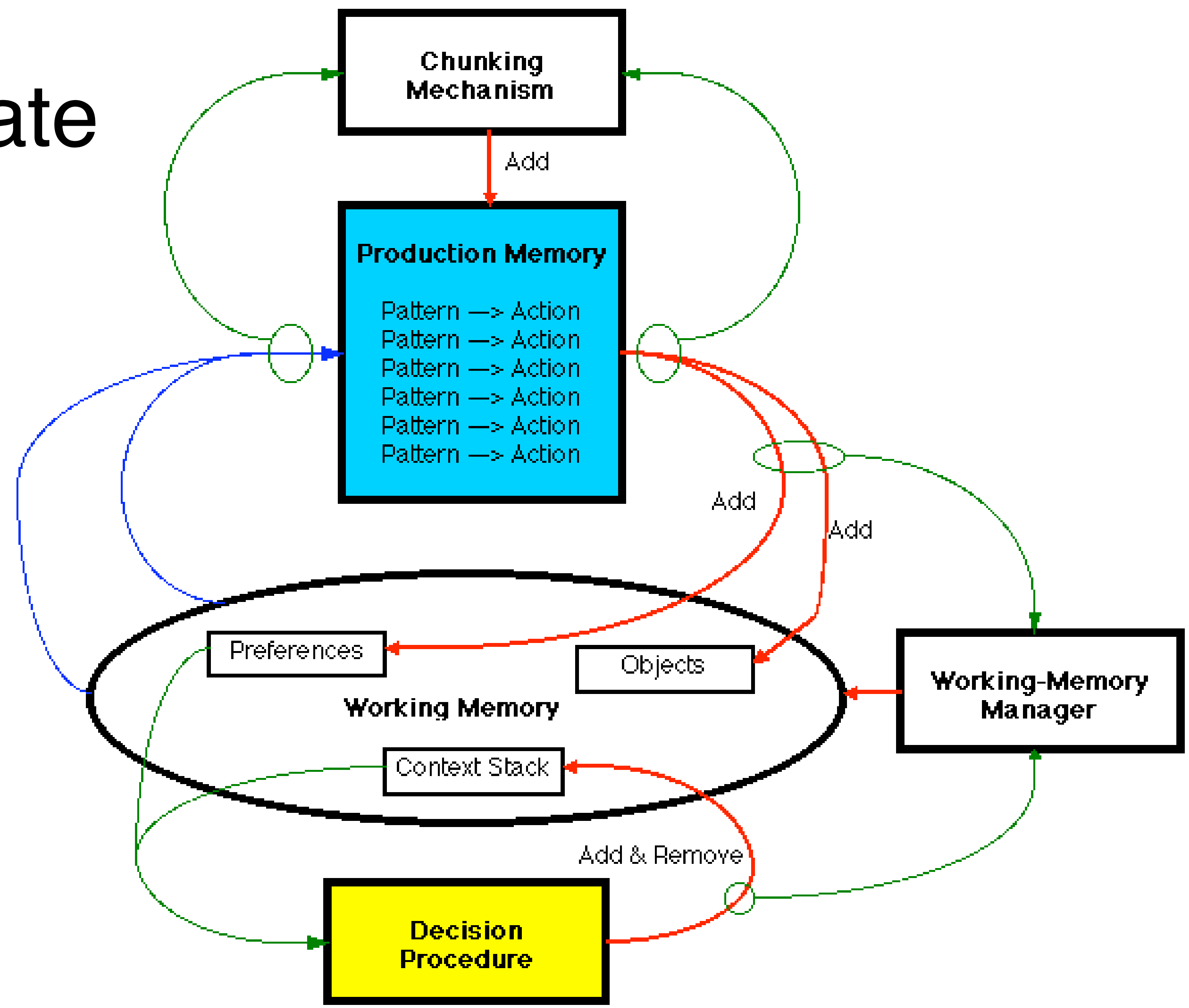
- From sensing to action (c.f. Skinner; conditioning; Witkowski 2007.)
- These work -- basic component of intelligence.
- The problem is choice (search).
    - Requires an arbitration mechanism.

Production-Based Architectures

- Expert Systems: allow choice of policies, e.g. recency, utility, random.
- SOAR: problem spaces (from GPS), impasses, chunk learning.
- ACT-R: (Bayesian) utility, problem spaces (reluctantly, from SOAR/GPS.)

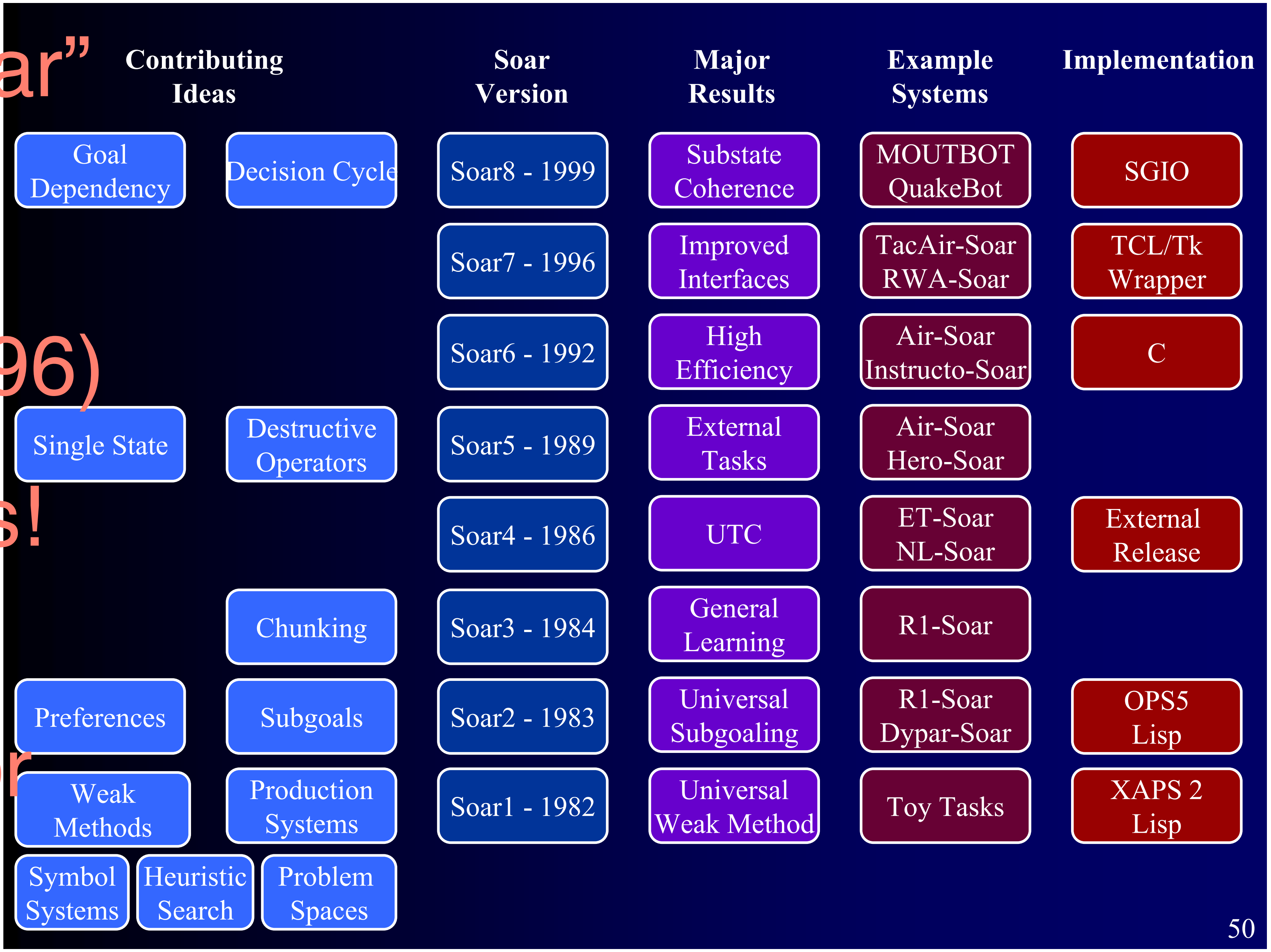
Soar

- Productions operate on predicate database.
- If conflict, declare impasse, reason (search).
- Remember resolution: chunk



Soar

- Soar has serious engineering.
- “Evolution of Soar” is my favorite paper (Laird & Rosenbloom 1996)
- Admits problems!
    - Not enough applications for human-like AI

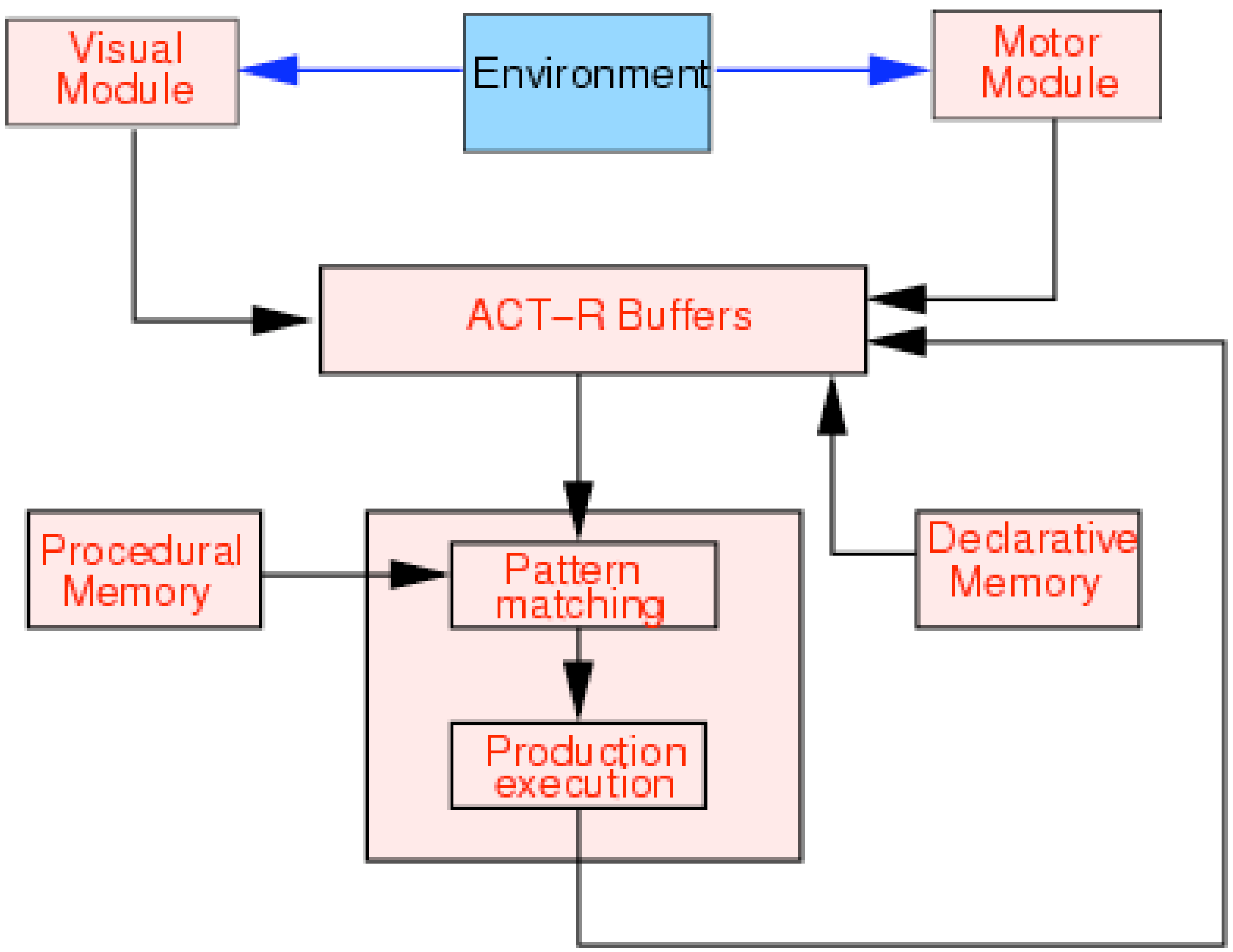


Architecture Lessons (from CMU)

- An architecture needs:
  - action from perception, and
  - further structure to combat combinatorics.
- Dealing with time is hard.

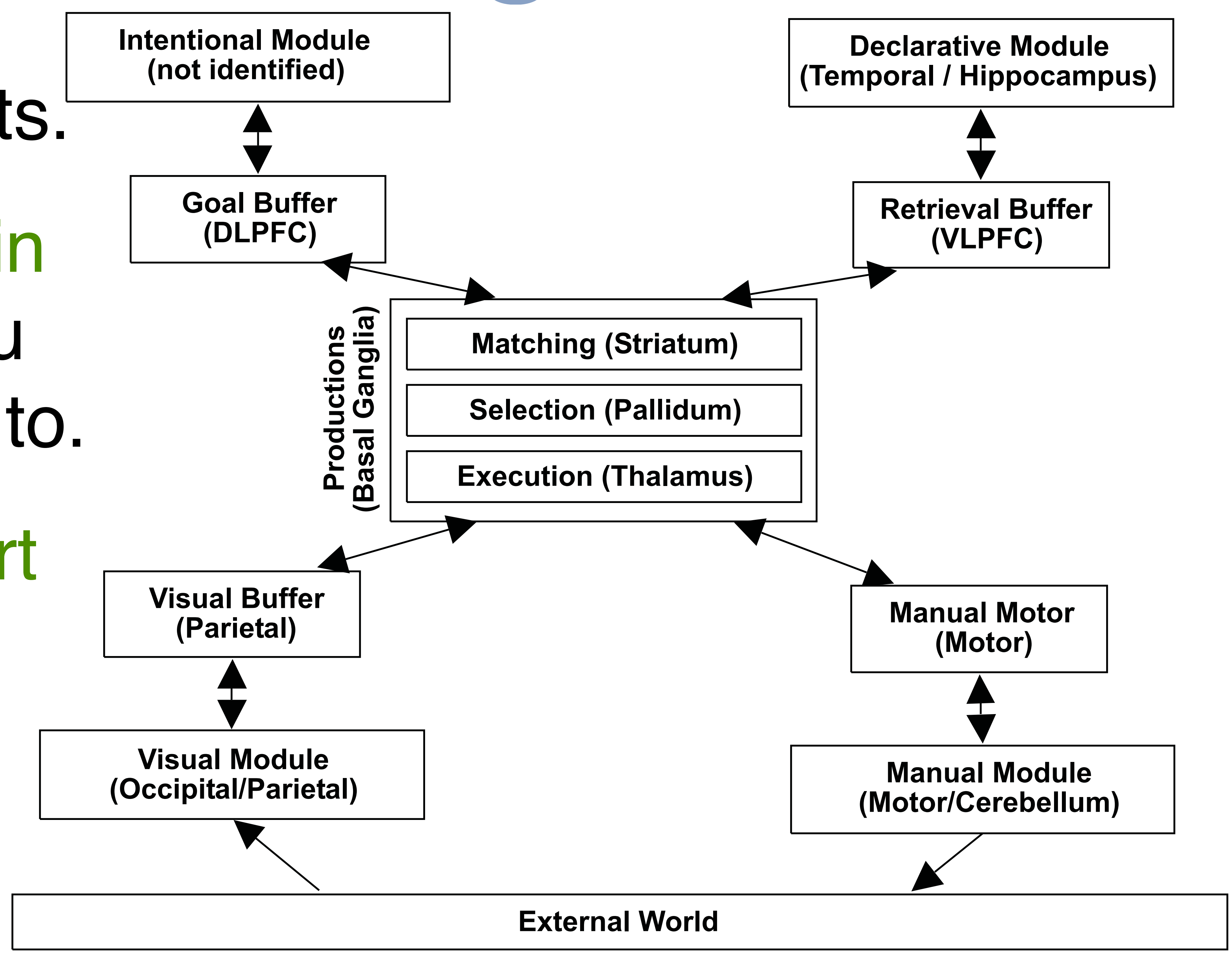
ACT-R

- Learns (& executes) productions.
- For arbitration, rely on (Bayesian probabalistic) utility.
- Call it implicit knowledge.



ACT-R Research Programme

- Replicate lots of Cognitive Science results.
- See if the brain does what you think it needs to.
- Win Rumelhart Prize (John Anderson, 2000).



Architecture Lessons (from CMU)

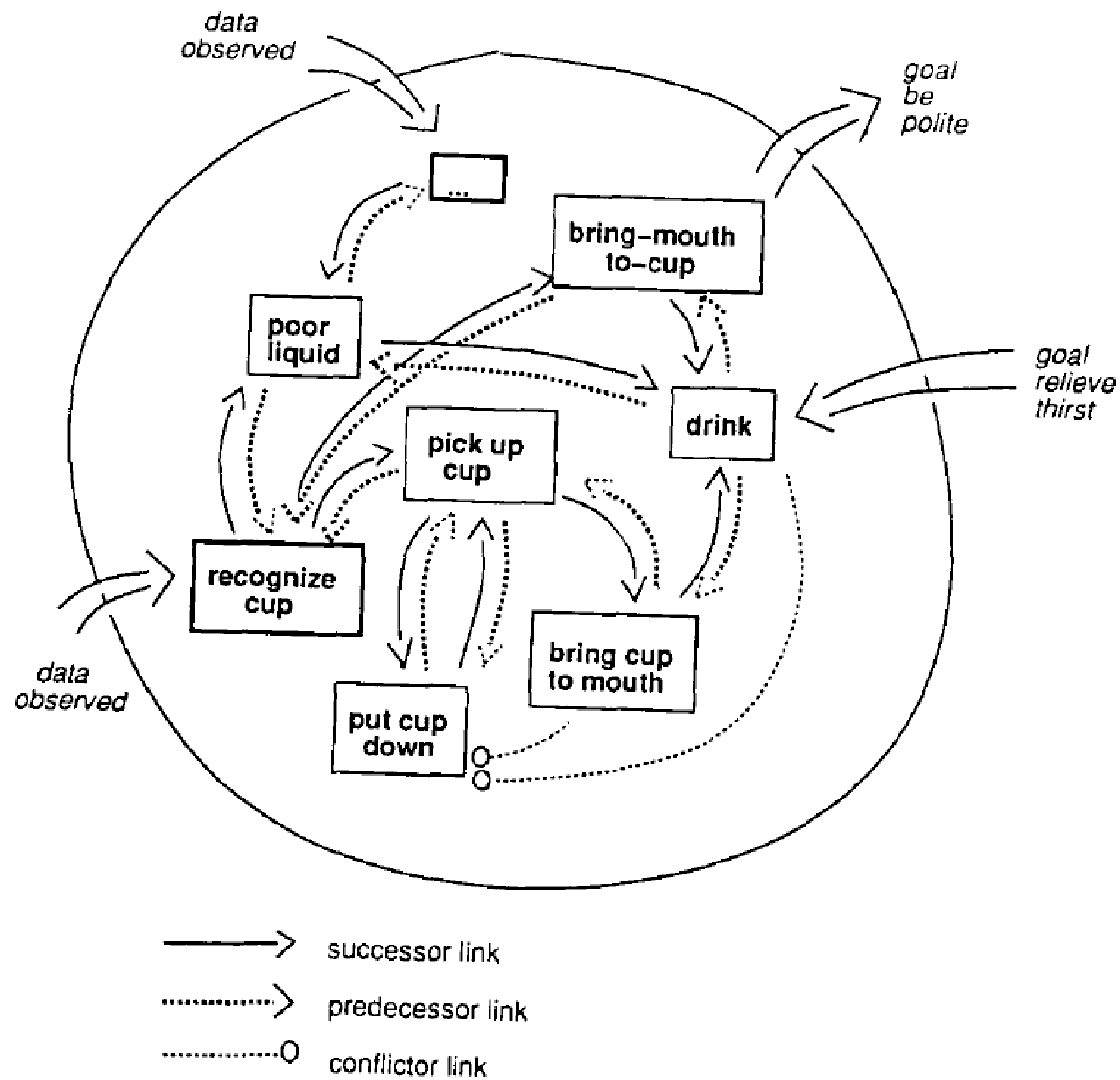
- Architectures need productions and problem spaces.
- Real-time is hard.
- Being easy to use can be a win.

Architecture Lessons (from CMU)

- Architectures need productions and problem spaces.
- Real-time is hard.
- Being easy to use can be a win.

Spreading Activation Networks

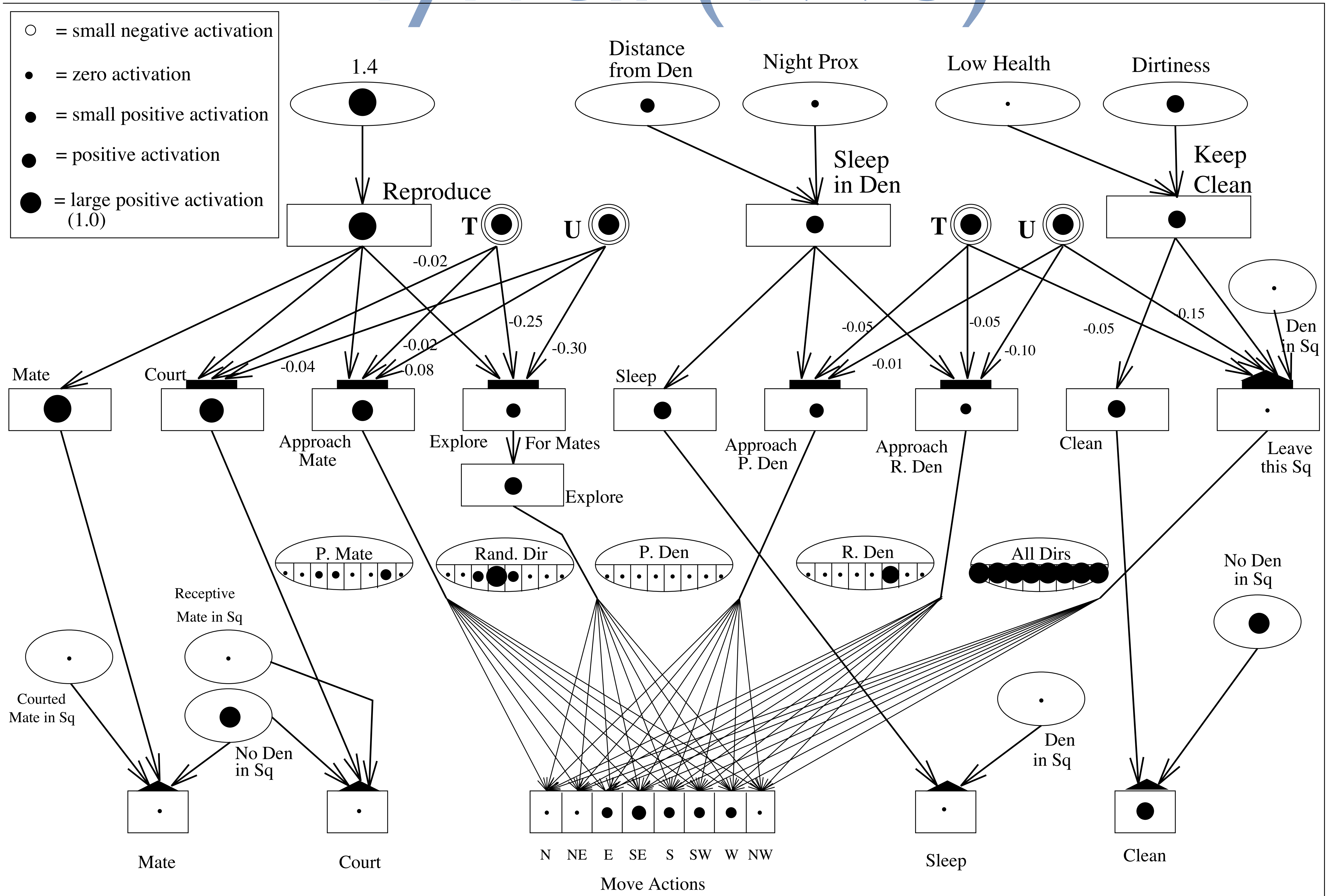
- "Maes Nets" (Adaptive Neural Arch.; Maes 1989)
- Activation spreads from senses and from goals through net of actions.
- Highest activated action acts.



Spreading Activation Networks

- Sound good:
  - easy
  - brain-like (priming, action potential).
  - Still influential (Franklin 2000, Shanahan 2006).
- Can't do full action selection:
  - Don't scale; don't converge on comsumatory acts (Tyrrell 1993).

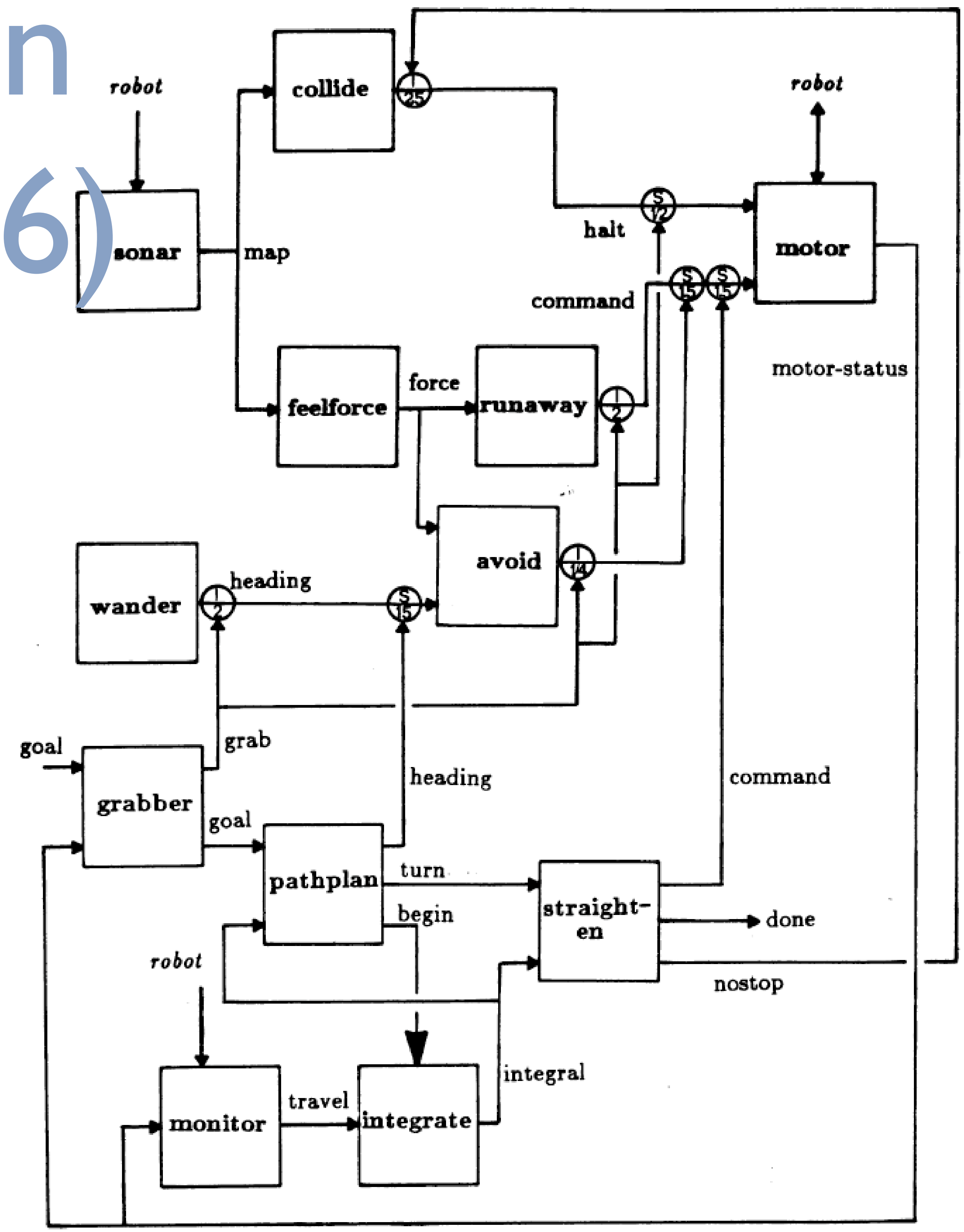
Tyrrell (1993)



Extended Rosenblatt and Payton Free-Flow Hierarchy

Subsumption (Brooks 1986)

- Emphasis on sensing to action (via Augmented FSM).
- Very complicated, distributed arbitration.
- No learning.
- Worked.

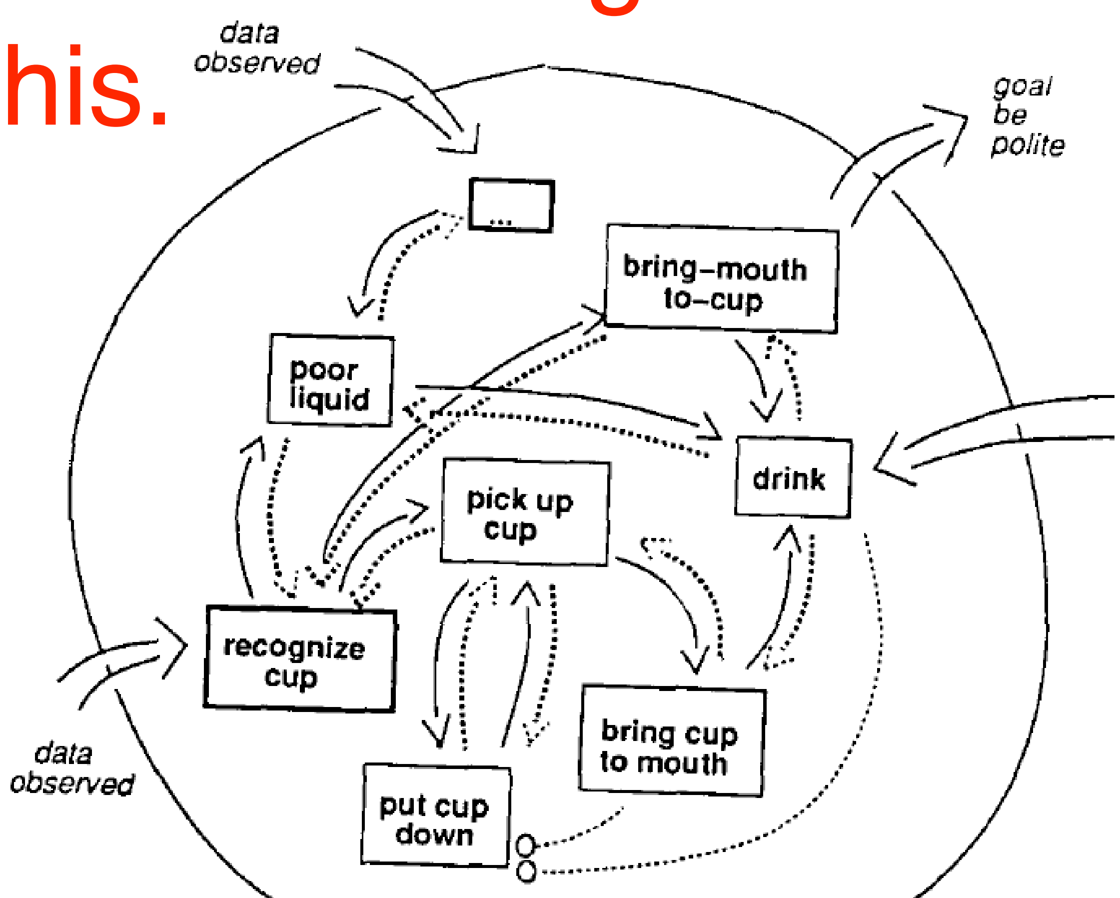


Architecture Lessons (Subsumption)

- Action from perception can provide the further structure -- modules (behaviors).
  - Modules also support iterative development / continuous integration.
- Real time should be a core organizing principle -- start in the real world.
- Good ideas can carry bad ideas a long way (no learning, hard action selection).

Architecture Lesson?

- Goals ordering needs to be flexible.
- Maybe spreading activation is good for this.



A Robust Layered Control System for a Mobile Robot

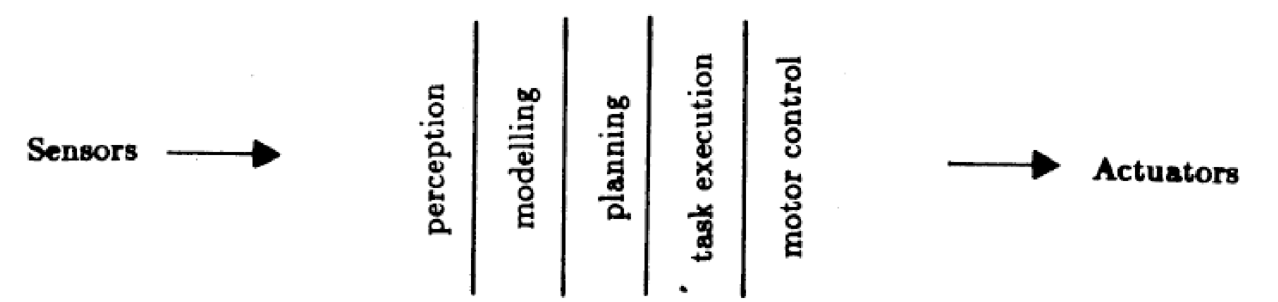
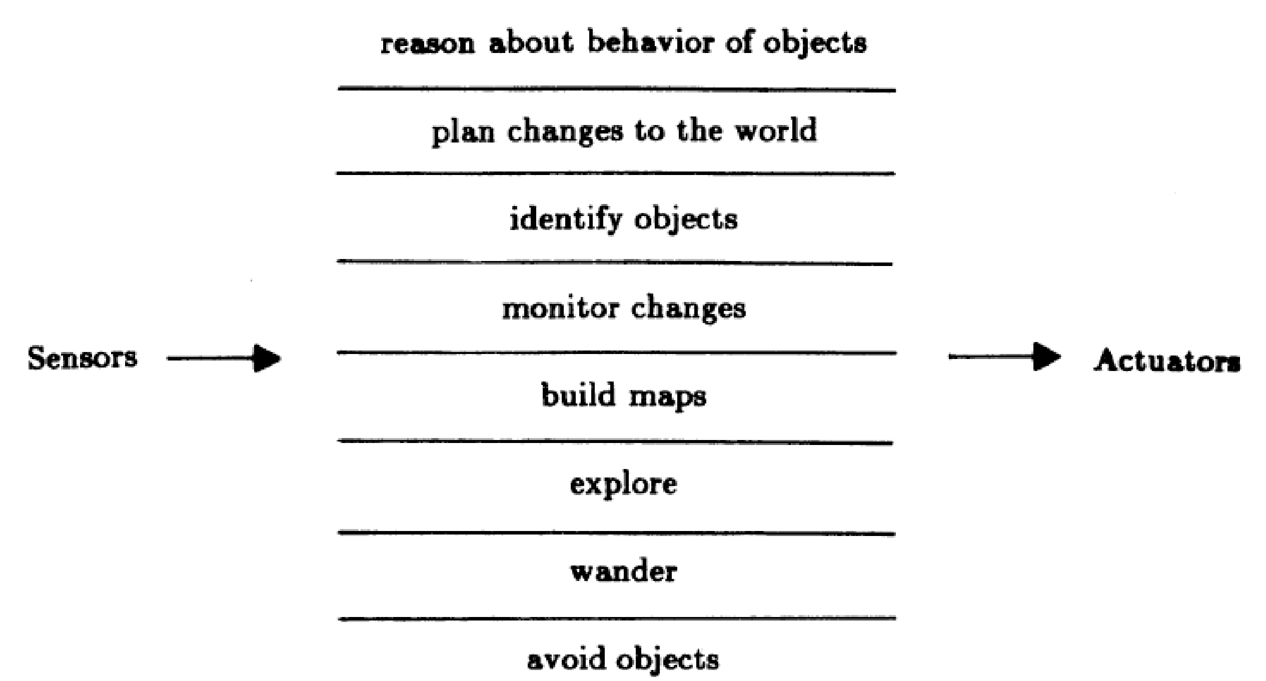
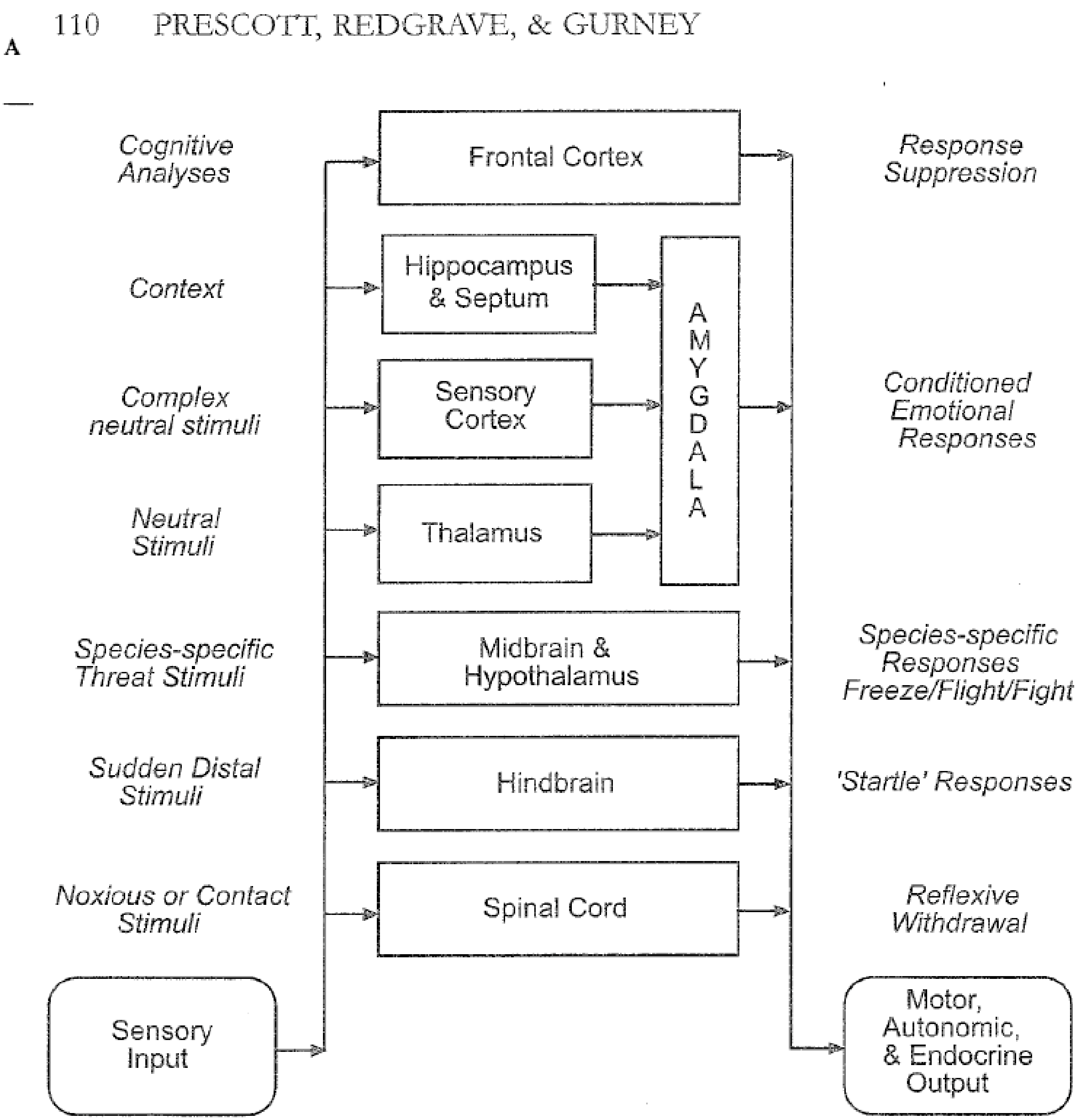
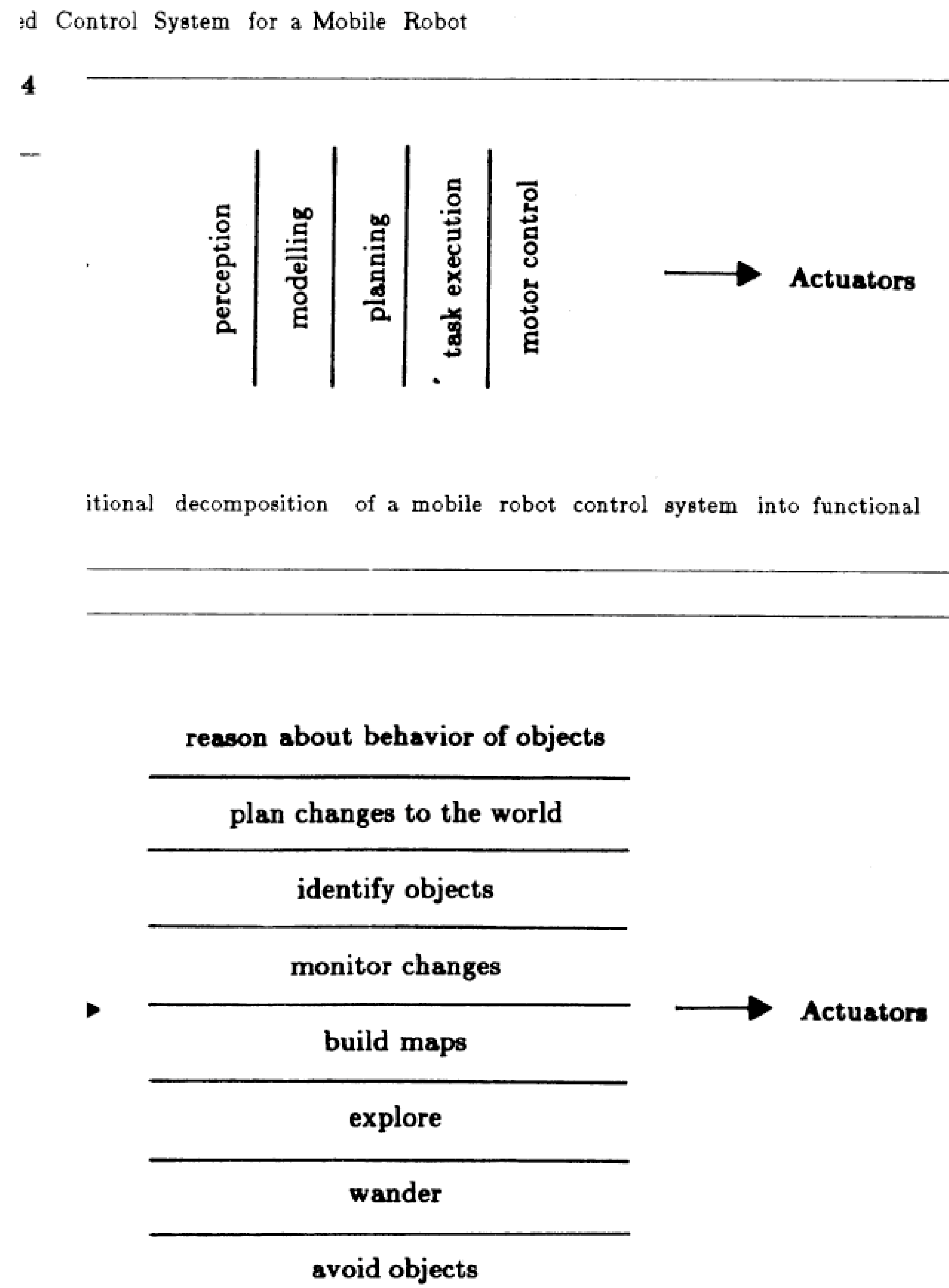


Figure 1. A traditional decomposition of a mobile robot control system into functional modules.



SA: Layers vs. Behaviours





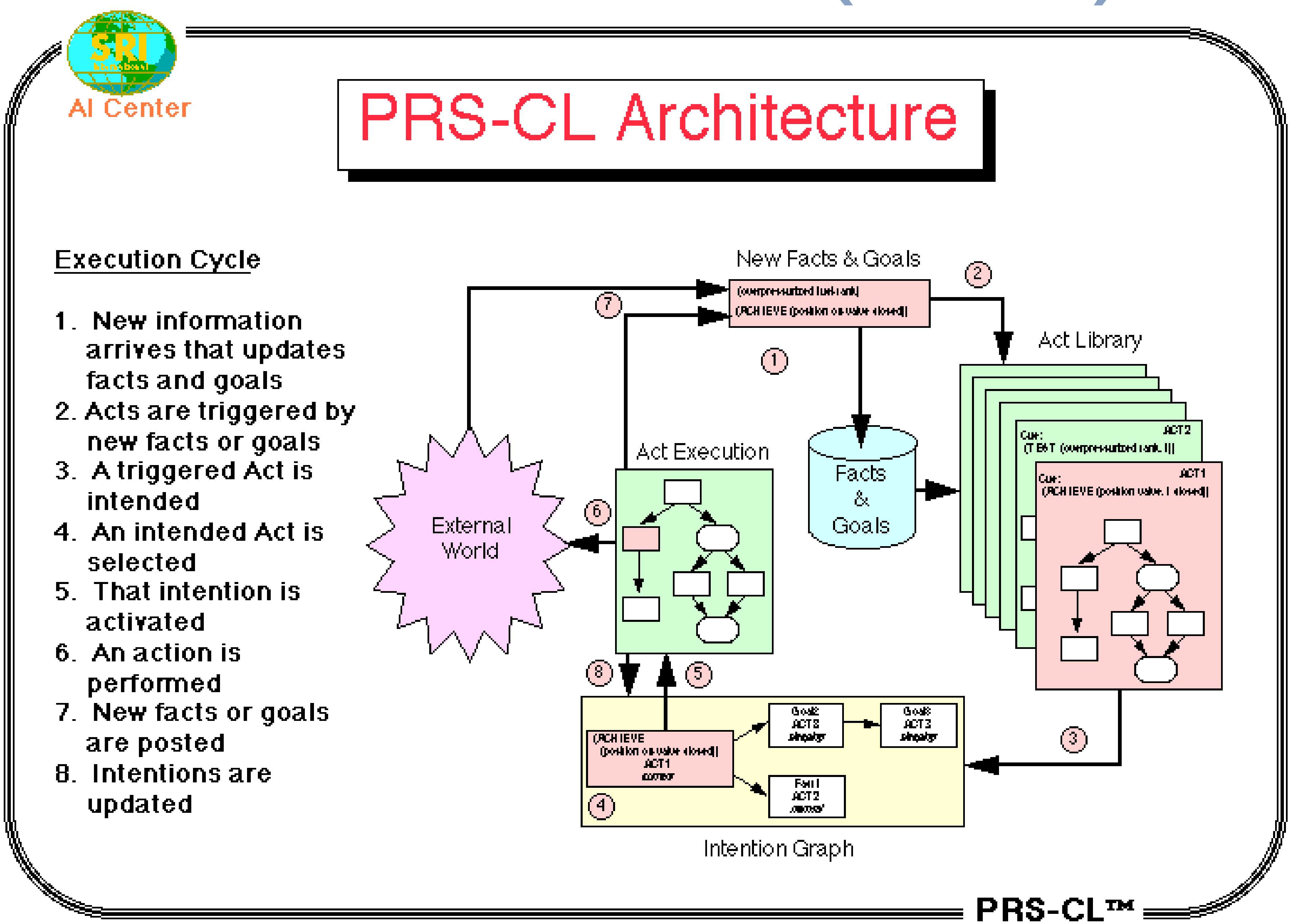
Layered *or* Hybrid Architectures

1. Incorporate behaviors/modules (action from sensing) as "smart" primitives.
2. Use hierarchical dynamic plans for behavior sequencing.
3. (Allegedly) some have automated planner to make plans for layer 2.

- Examples: Firby/RAPS/3T ('97); PRS (1992-2000); Hexmoore '95; Gat '91-98

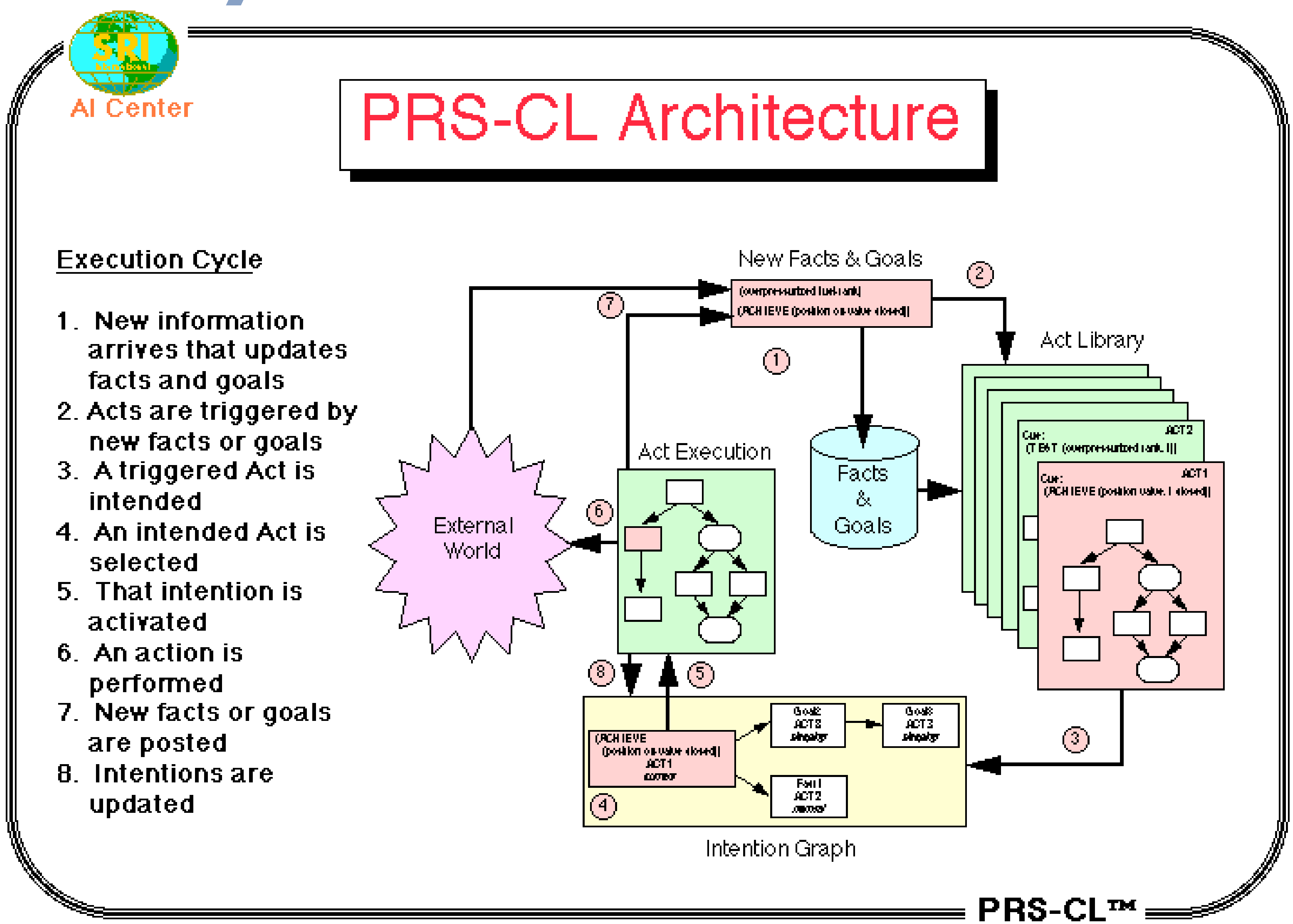
Belief, Desires, Intentions (BDI)

- *Beliefs*: Predicates
- *Desires*: goals & related dynamic plans
- *Intentions*: current goal



Procedural Reasoning System

- BDI
- And reactive (responds to emergencies by changing intentions.)
- Er... once or twice (Bryson ATAL 2000).



Architecture Lessons

- Structured dynamic plans make it easier to get your robot to do complicated stuff.
- Automated planning (or for Soar, chunking/learning) is seldom actually used.
- To facilitate that automated planning, modularity is often compromised.

Soar as a 3LA

- J. Laird & P. Rosenbloom, “The Evolution of the Soar Cognitive Architecture”, *Mind Matters,* D. Steier and T. Mitchell eds., 1996.

CogAff

- Reflection on Top.
- Sense & Action separated!
- (Davis & Sloman 1995)

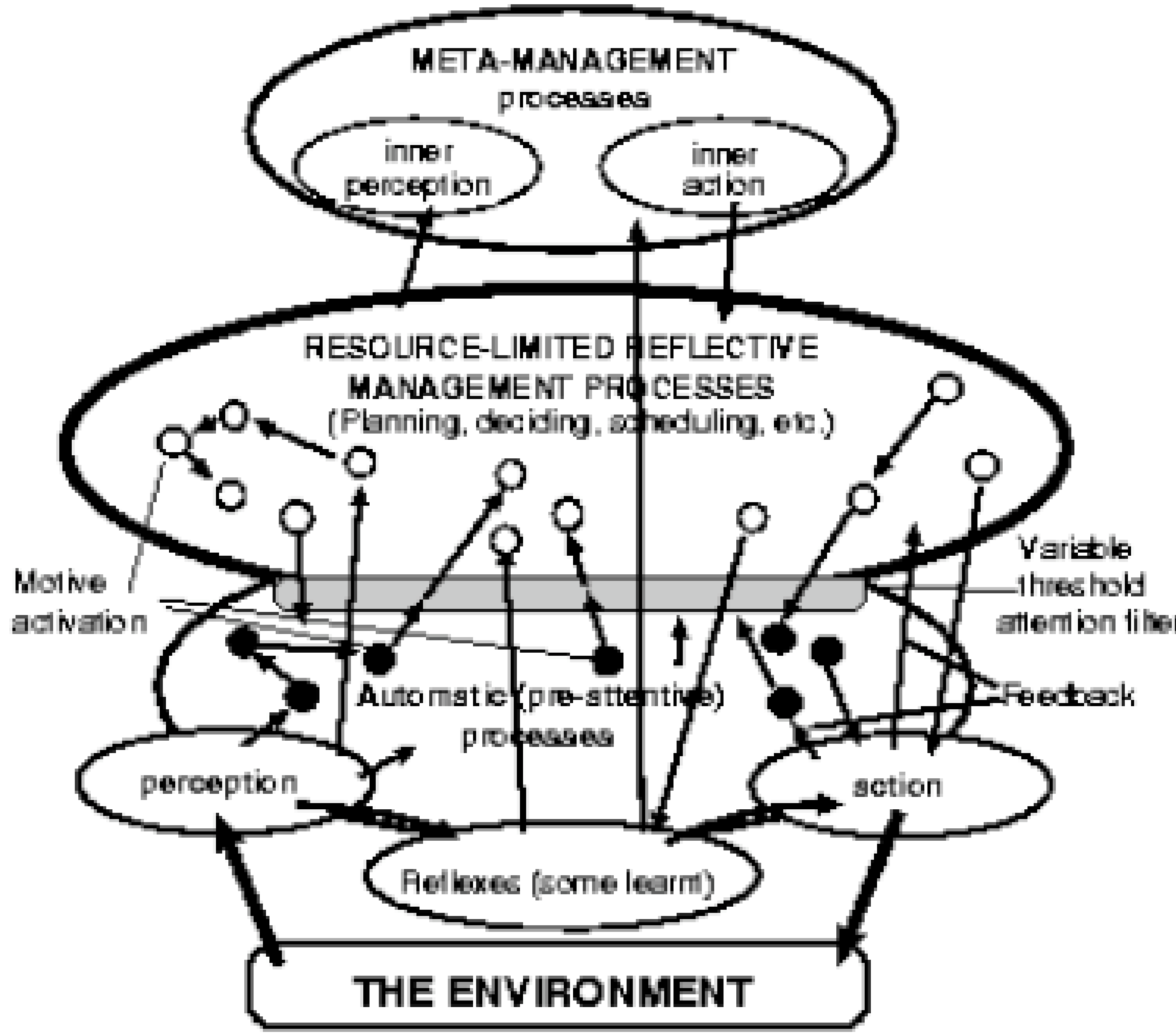
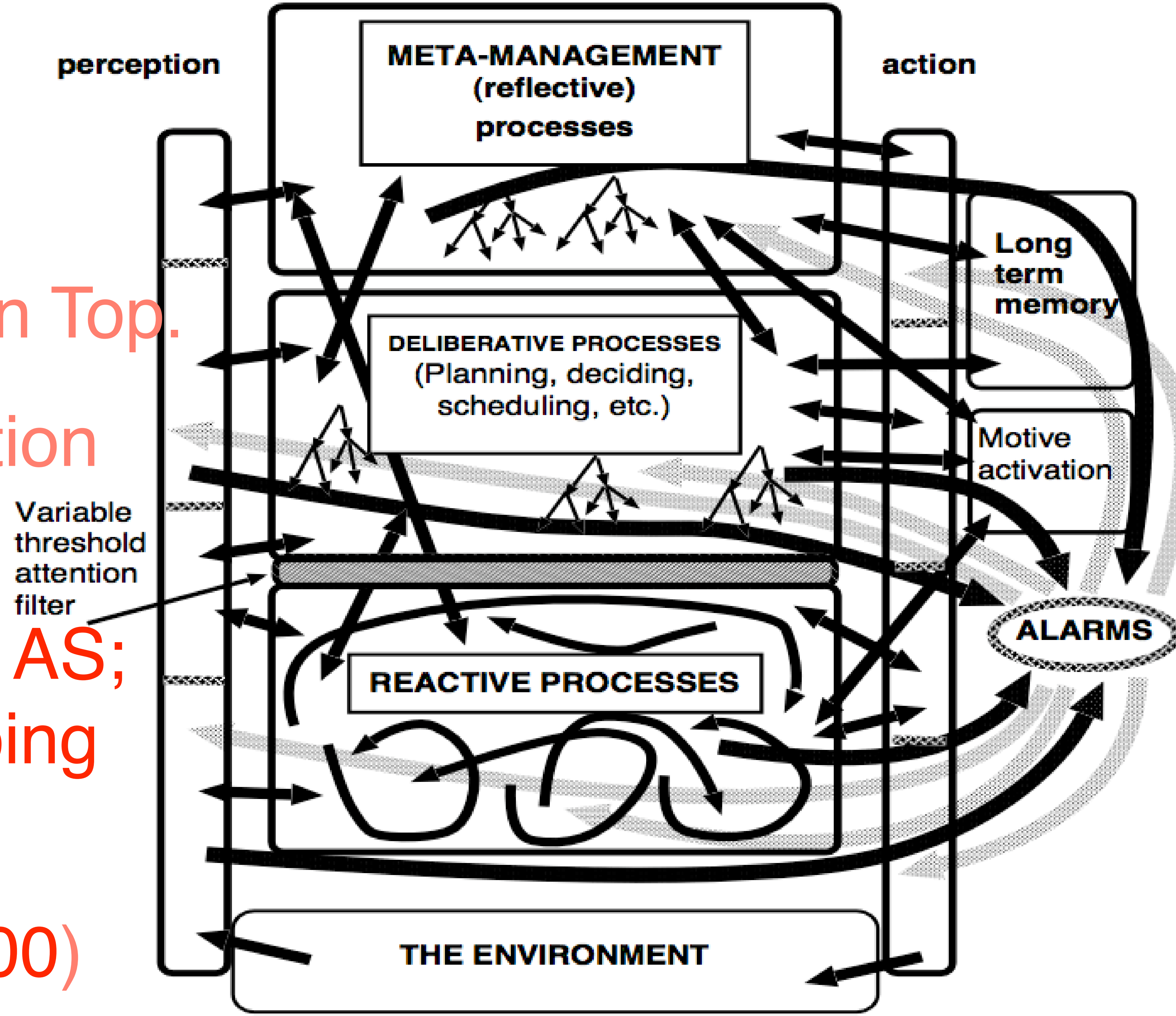


Figure 1 Towards an Intelligent Agent Architecture

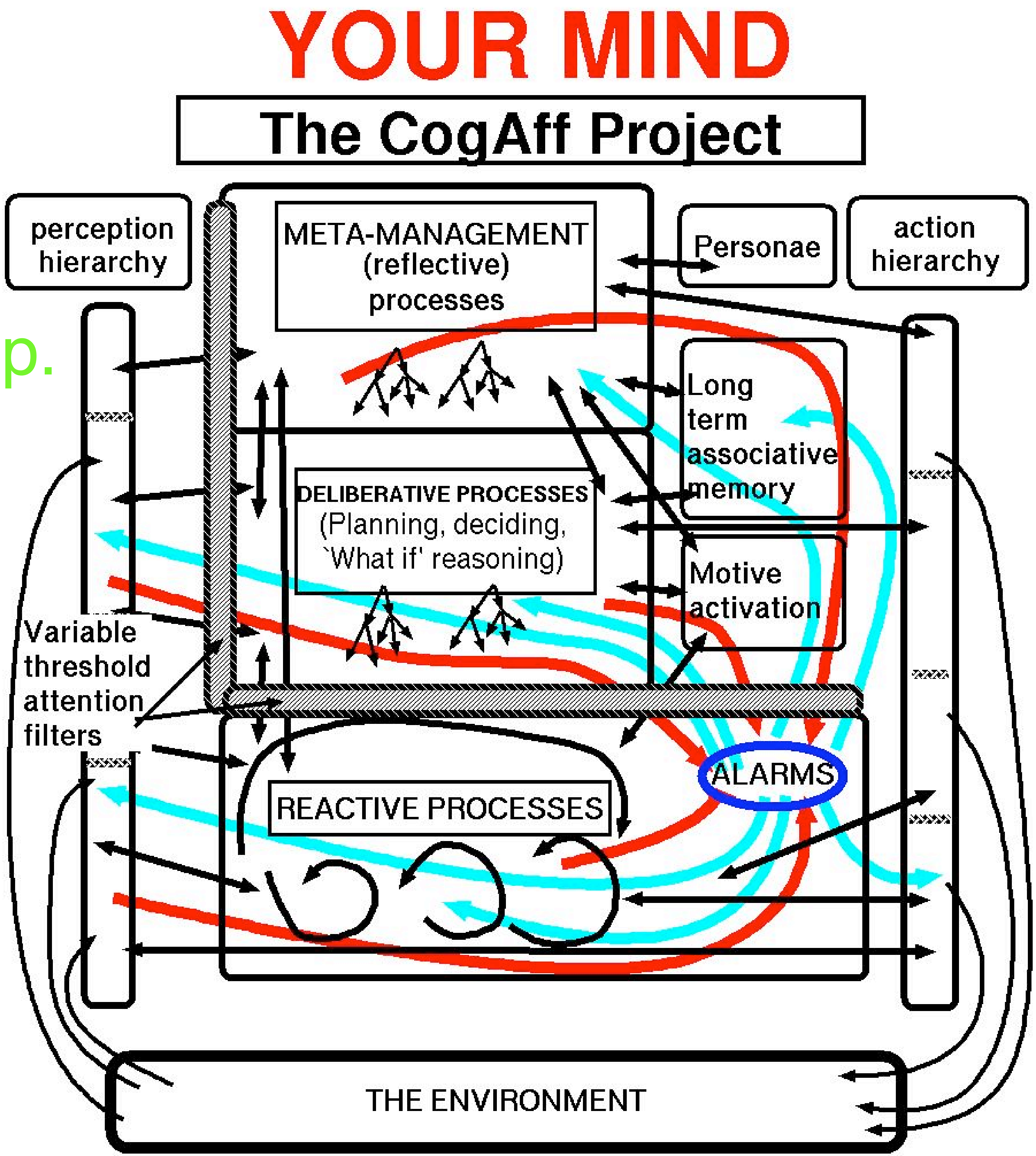
CogAff

- Reflection on Top.
- Sense & Action separated!
- Hierarchy in AS; Goal Swapping (Alarms).
- (Sloman 2000)

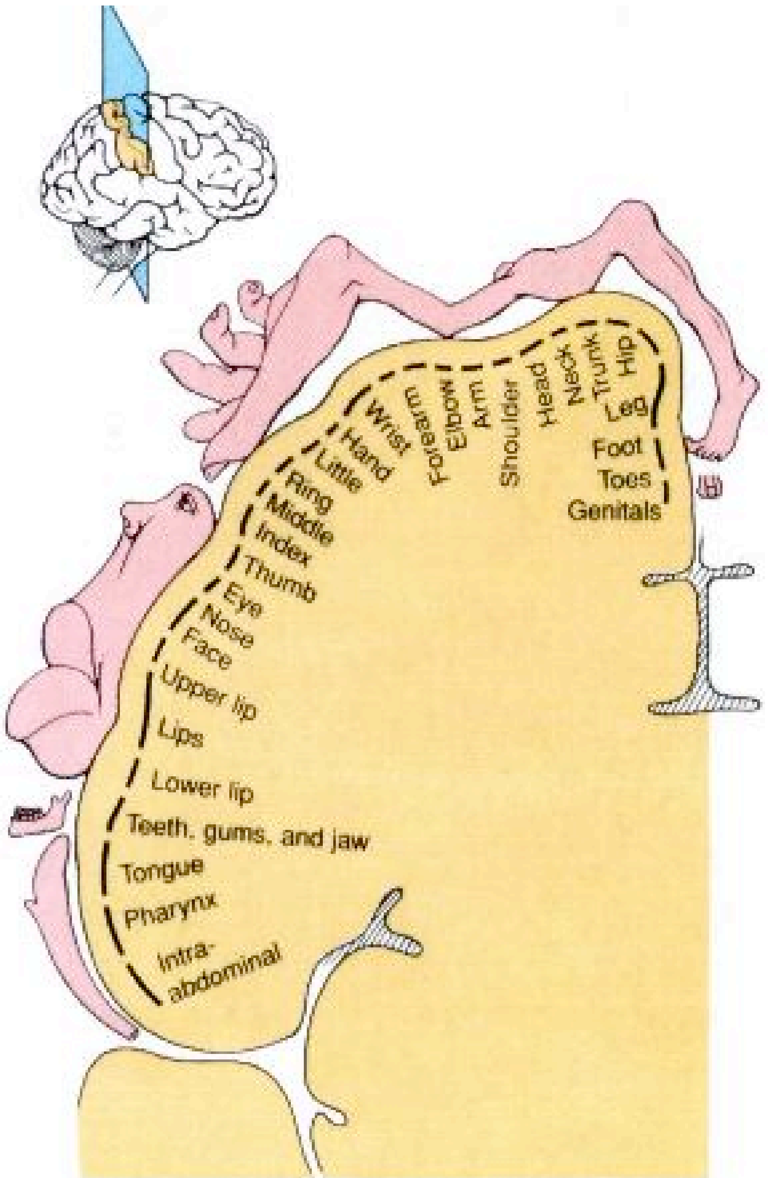


CogAff

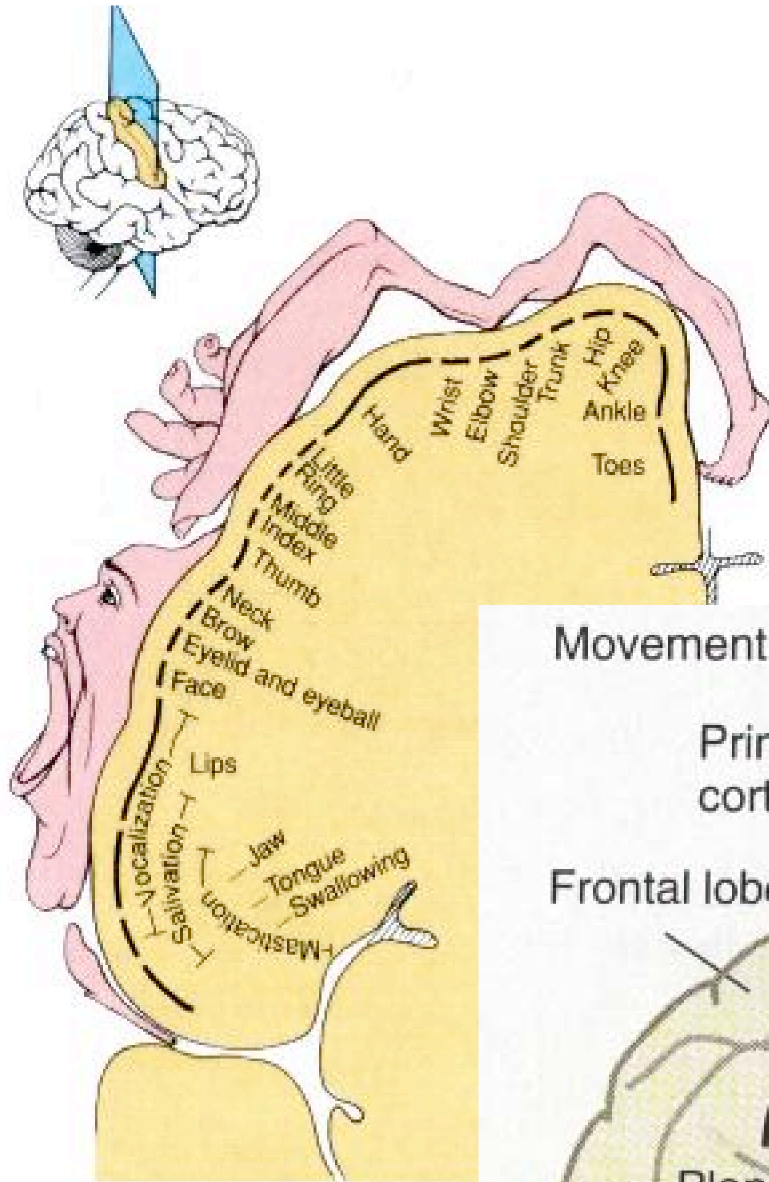
- Reflection on Top.
- Sense & Action separated!
- Hierarchy in AS, Goal Swapping (now reactive).
- Current Web



Separate Sense & Action

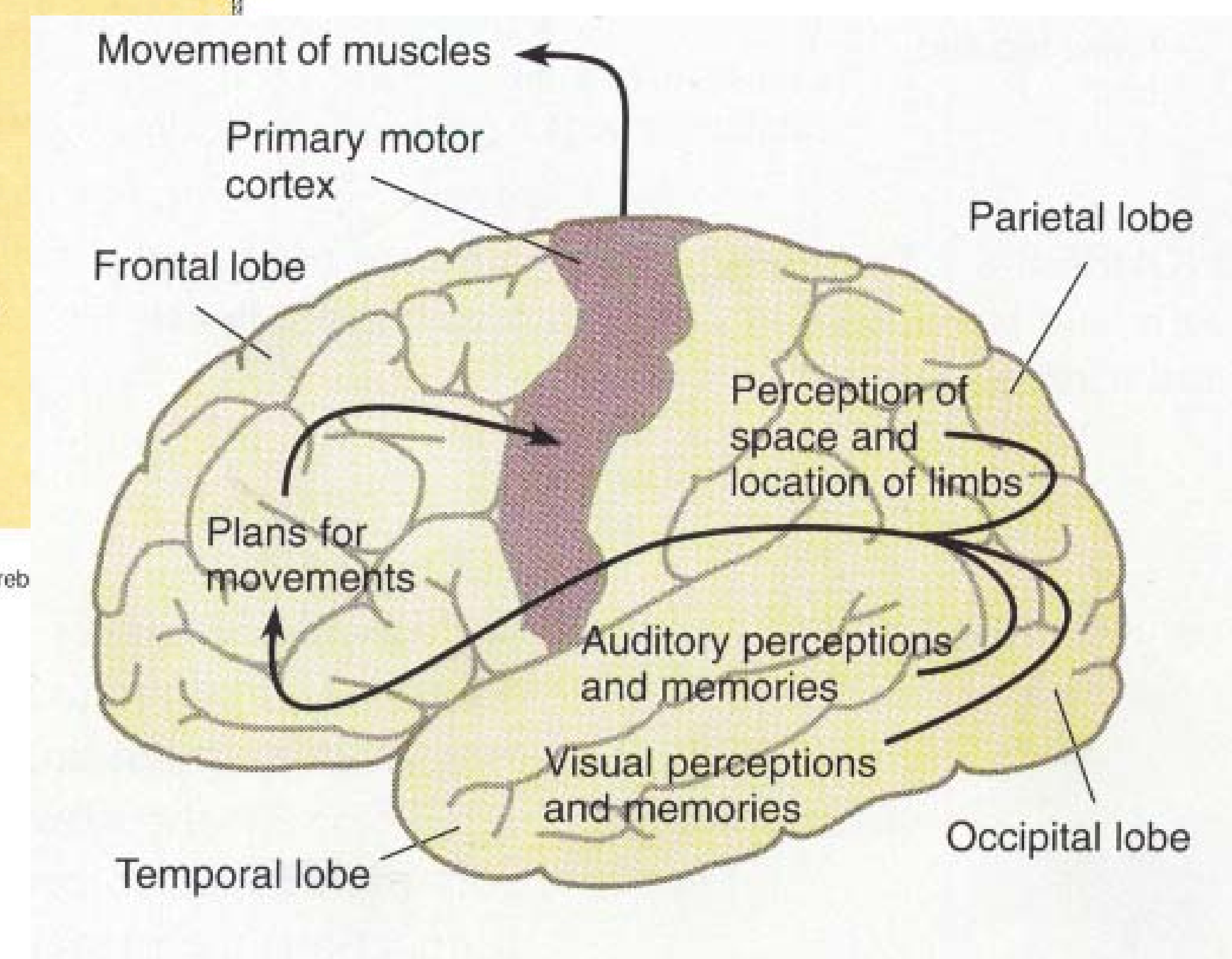


(a) Somatosensory cortex in right cerebral hemisphere



(b) Motor cortex in right cereb

- Something we higher mammals do.
- Central Sulcus



Chance for Cognition?

(pictures from Carlson)

Architecture Lessons (CogAff)

- Maybe you don't really want productions as your basic representation -- you may want to come between a sense and an act sometimes.
- Aaron thinks about a lot more cognitive stuff than I do.

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
  - SOAR/ACT-R, ANA (Maes Nets), Subsumption, BDI/PRS, CogAff, Brains
- Behavior Oriented Design

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design

Outline

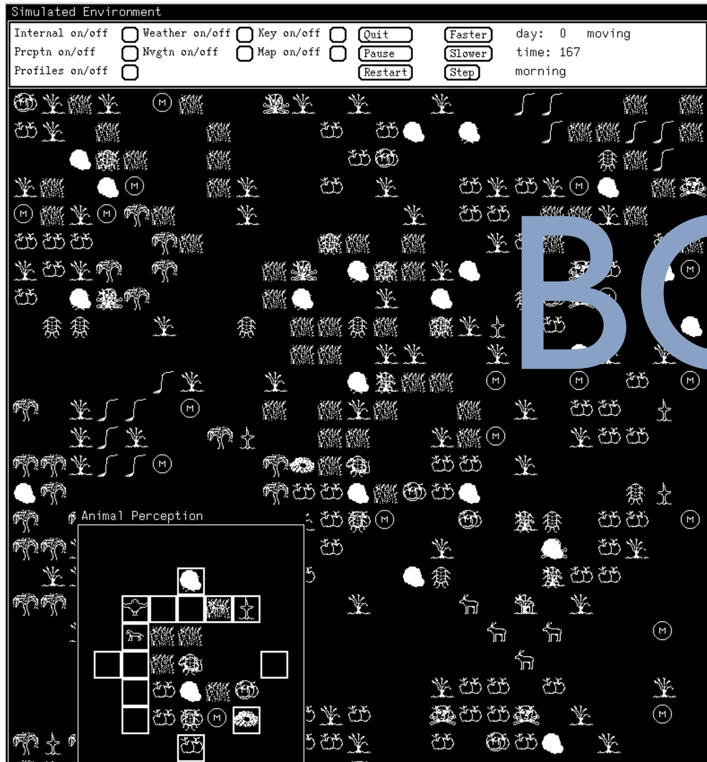
- Introduction
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design
    - Conclusion / Recommendations

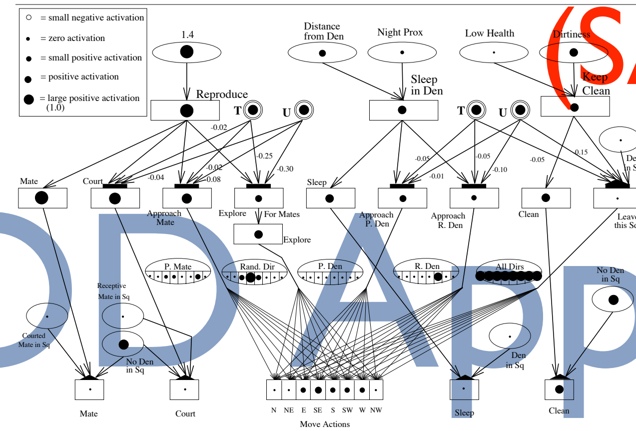
Behavior Oriented Design

- All search (learning, planning) is done within modules with specialized representations.
- Specialized representations promote reliability of search; also determine decomposition.
- Modules provide perception, action, memory. Arbitration via hierarchical dynamic plans.
- Iterative / agile test & development cycle.

(Bryson 2001, 2003)

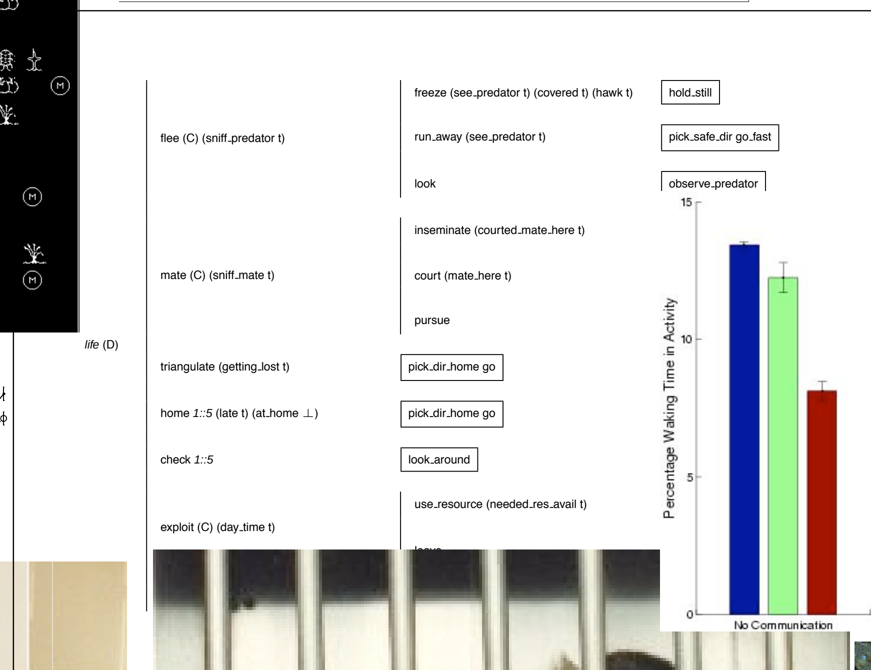
BOD Applications

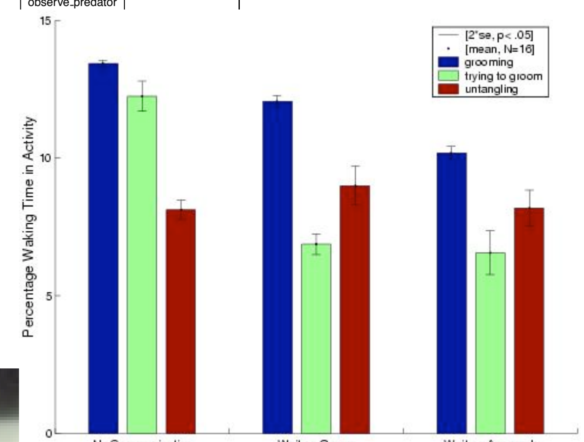


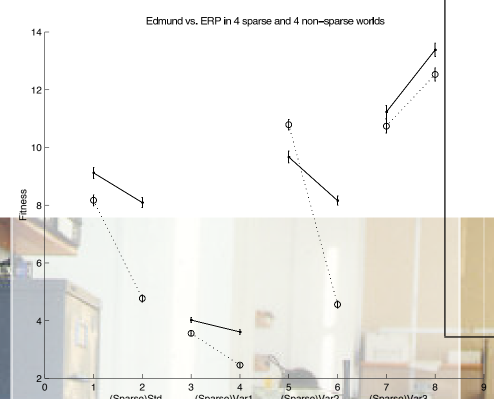


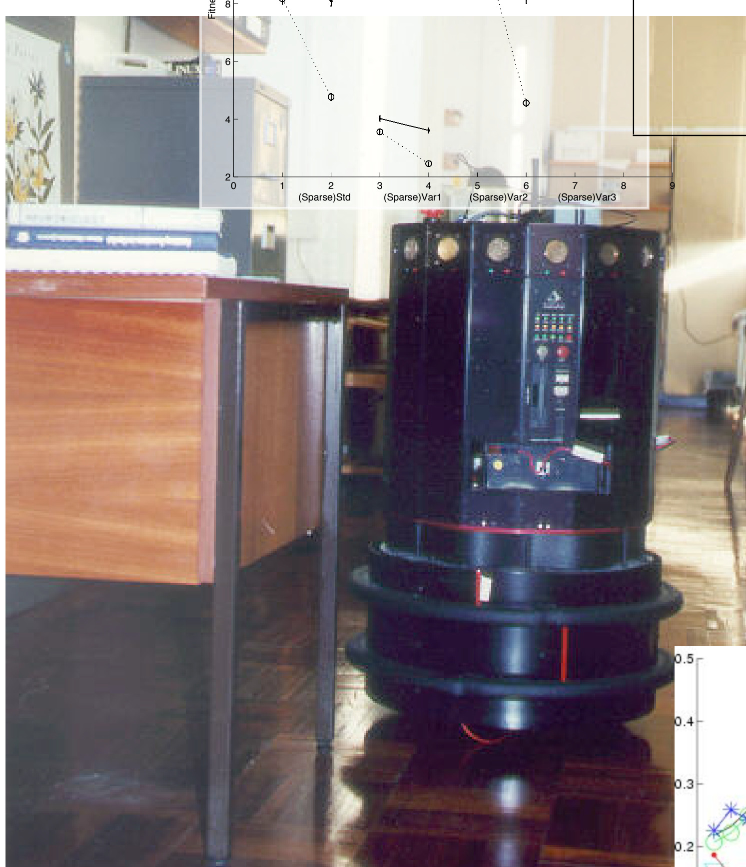
(SAB 2000)

(VR(J) 2000)





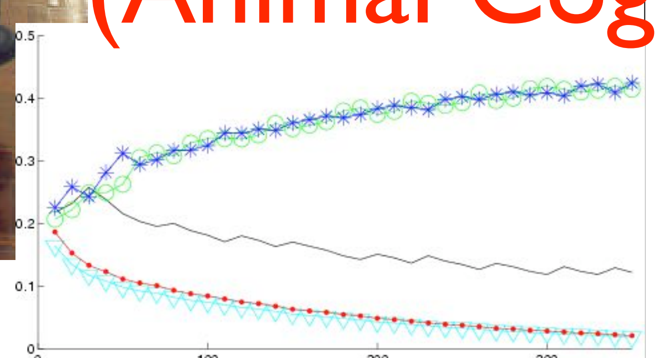


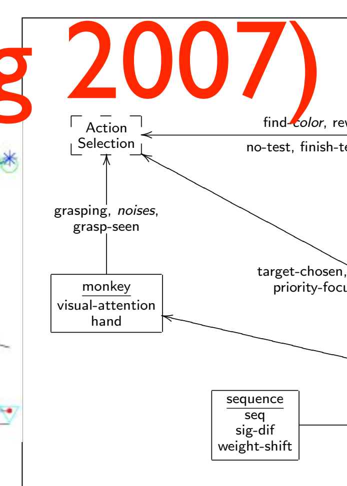


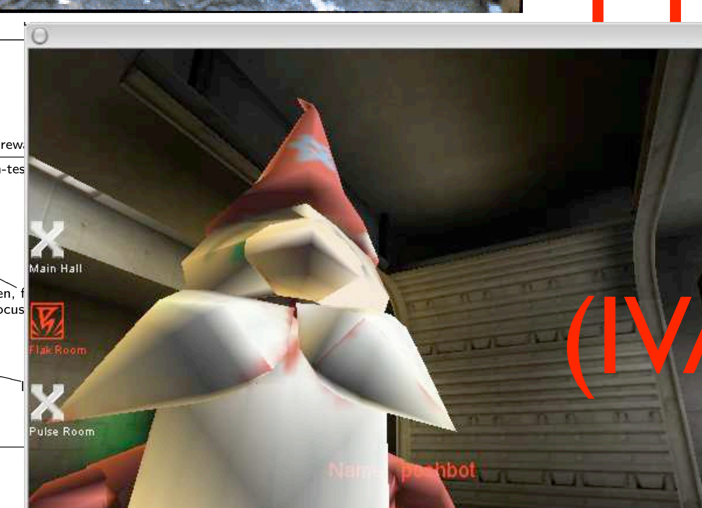
(ATAL 1997)

(WRAC 2003, PTRS B 2007)

(Animal Cog 2007)







(IVA 2005)

What I Learned from Robots

1. Perception is hard (explains the brain).
   - Lead to specialized representations encapsulated in modules; my method of behavior-module decomposition.
2. Discrete action selection is compatible with continuous acting, provided the primitive `acts' alter ongoing behaviour supported by modules.
   - e.g. motor act sends target velocity, not vector;
   - multiple || devices/modules e.g. speech, motion.

Modularity is not Enough

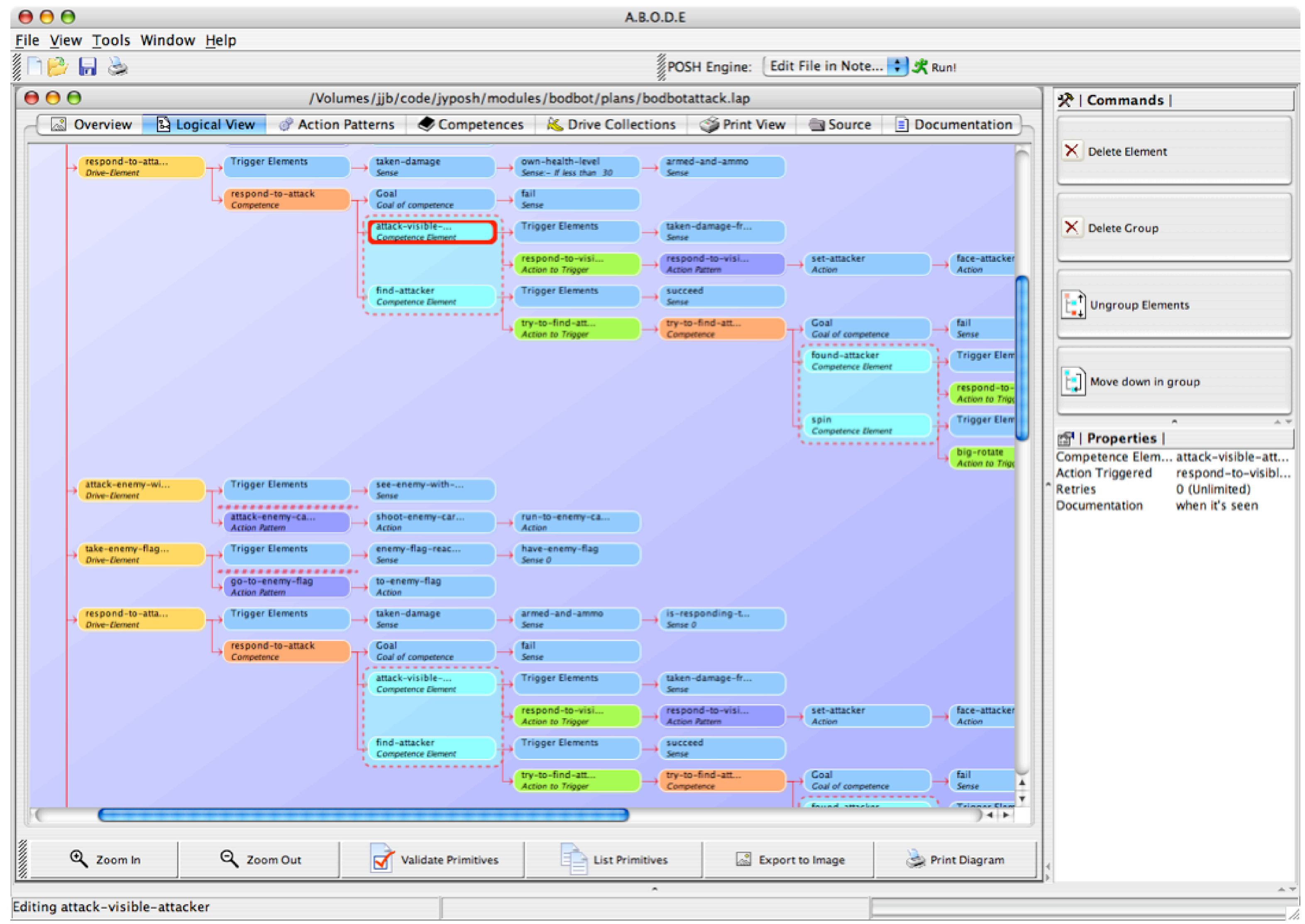
Get Fuzzy (Conley 2006)

BOD Action Selection

Parallel-rooted, Ordered, Slip-stack Hierarchical (POSH) action selection:

- Some things need to be checked at all times: drive collection.
- Some things only need considering in particular context: competences.
- Some things reliably follow from others: action patterns.

POSH plan in ABODE (for UT: Capture the Flag)



- Advanced BOD Environment.
- Initial development funded by industry.

Current Work

- Drive / Emotion level work with latching not adequate for reprioritizing goals.
  - Thinking about interrupts (Brom 2007; Norman & Shallice 1988) -- ‘spreading activation’ just for goals.
- Trying to make ABODE a real IDE.
- Modelling primate social behaviour.

Architecture Lessons

- Modularity: problem spaces, combat combinatorics, allow locally-optimal representations.
  - Should use ordinary (OO) code (arbitrarily powerful but also access to primitives.)
- Hierarchical action selection for arbitration.
- Dedicated, high-frequency goal / attention switching, compensates for hierarchical AS.
- Agile development, refactoring (Beck 2000).

Architecture Lessons: What Doesn't Work

- Calling Subsumption or ANA an architecture.
- Development methodologies:
  - Describe ontologies / representations;
  - Recommend development strategies.

We need to help the average programmer.

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design
  - Conclusion / Recommendations

Outline

- Introduction
- A Brief History of AI Cognitive Architectures
- Behavior Oriented Design

AI Architectures

or

State Requirements for Human-Like Action Selection

Joanna J. Bryson
Artificial models of natural Intelligence, University of Bath
Konrad Lorenz Institute for Evolution and Cognition Research

BOD Development Cycle

1. Initial decomposition ⇒ specification.
2. Scale the system.
   i. Code one behavior and/or plan.
   ii. Test and debug code (test earlier plans).
   iii. Simplify the design.
3. Revise the specification.
4. Iterate.

BOD Development Cycle

1. Initial decomposition ⇒ specification.
2. Scale the system.
   i. Code one behavior and/or plan.
   ii. Test and debug code (test earlier plans).
   iii. Simplify the design.
3. Revise the specification.
4. Iterate.

1. Specify (high-level) what the agent will do.
2. Describe activities as sequences of actions. competences and action patterns
3. Identify sensory and action primitives from these sequences.
4. Identify the state necessary to enable the primitives, cluster primitives by shared state. behavior modules
5. Identify and prioritize goals / drives. drive collection
6. Select a first (next) behavior to implement.

BOD Development Cycle

1. Initial decomposition ⇒ specification.
2. Scale the system.
   1. Code one behavior and/or plan.
   2. Test and debug code (test earlier plans).
   3. Simplify the design.
3. Revise the specification.
4. Iterate.

BOD Development Cycle

1. Initial decomposition ⇒ specification.
2. Scale the system.
   i. Code one behavior and/or plan.
   ii. Test and debug code (test earlier plans).
   iii. Simplify the design.
3. Revise the specification.
4. Iterate.

Simplify the Design

Use the simplest representations.

- Plans:
  - primitives, action patterns, competences.
  - drives only if need to always check.
- Behavior modules / memory:
  - none, deictic, specialized, general.

(Bryson, AgeS 2003)

Simplify the Design

Trade off representations: plans vs. behaviors

- Use simplest plan structure unless redundancy (split primitives for sequence, add variable state in modules).
- If competences too complicated, introduce primitives or create more hierarchy.
- Split large behaviors, use plans to unify.
- All variable state in modules (deictic).

(Bryson, AgeS 2003)

life (D)

- untangle (tangled?) — untangle
- groom (C) (want-to-groom?)
  - (partner-chosen?) (aligned?) — notify groom
  - (being-groomed?) — choose-groomer-as-partner
  - (partner-chosen?) (touching?) — notify align
  - (partner-chosen?) — notify approach
  - ($\top$) — choose-partner
- receive (being-groomed?) — tolerate-grooming
- explore (C) (want-novel-loc?)
  - (place-chosen?) (there-yet?) — lose-target
  - (place-chosen?) — explore-that-a-way
  - ($\top$) — choose-explore-target
- wait ($\top$) — wait

New Outline

- Introduction to Cognition (16 min)
- History of AI Architectures (24 min)
  - Make point about isomorphism of learning & search here.
- Behavior Oriented Design (10 min)
  - mention new idea, relation to Shannahan.

Cognition Intro

- Do Cognition as Search -- do search stuff from this talk, then do the provided, required, open slide.
- Then talk about search at different time steps: Baldwin effect -> cultural / bio evolution, semantic / episodic memory.
- Mention Dennett's free-floating rationale, Tinburgen's ultimate vs. proximate cause.