WHAT'S AN AI TOOLKIT FOR?

Aaron Sloman
http://www.cs.bham.ac.uk/˜axs/
A.Sloman@cs.bham.ac.uk
School of Computer Science
The University of Birmingham

Including ideas from:

Brian Logan,

Riccardo Poli,

Luc Beaudoin,

Darryl Davis,

Ian Wright,

Peter Waudby

Jeremy Baxter (DERA),

Richard Hepplewhite (DERA),

And various students and colleagues

OUR SIM_AGENT TOOLKIT IS AVAILABLE ONLINE

IN THE BIRMINGHAM POPLOG FTP DIRECTORY

ftp://ftp.cs.bham.ac.uk/pub/dist/poplog/

IS IT POSSIBLE TO PRODUCE *ONE* TOOLKIT WHICH MEETS ALL REQUIREMENTS, AND IF NOT WHY NOT?

We need to consider different sorts of uses of toolkits:

BOTH

Engineering goals such as producing intelligent robots, software systems, and symbiotic human-machine systems
AND

Scientific goals such as understanding existing intelligent systems and also trying to understand the space of possible designs, natural and artificial.

Brian Logan's paper is concerned with classifying types of agent systems, whereas I am more concerned with classifying the issues that arise in developing agent systems, though obviously the two are closely related.

The development issues include:

- **What sorts of things need to be put together?**
- **How many different ways are there of putting things together?**
- **What are the reasons for choosing between them?**
- **Should individuals be designed, or self-adapted or evolved, or ...?**

ANSWERS WILL OBVIOUSLY DEPEND ON

(a) *what* **is being assembled, including how complex the individual agents are, what they have to interact with, etc.**

(b) How well specified the task is initially

(c) Whether further development work may be required once the system is up and running

(d) What sorts of testing will be required.

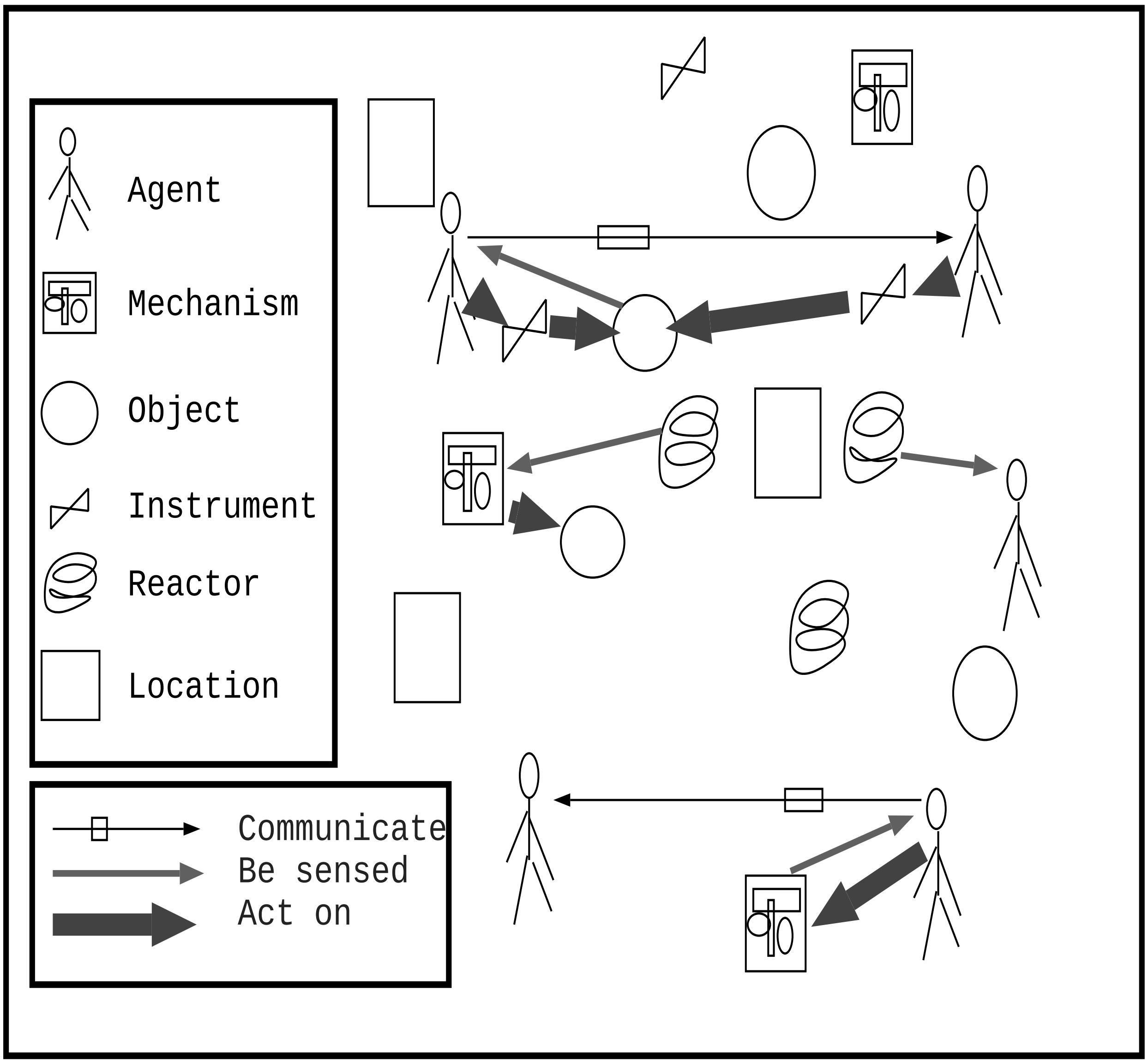
(e) Whether the objective is to produce a working tool, or to explore design issues and test theories, e.g. about humans or other animals.

So:

- **A general toolkit should not be committed to any particular architecture.**
- **It should support a range of design and development methodologies.**
- **It should allow the user to address tradeoffs between:**
    - **speed**
    - **ease of development and testing**
    - **flexibility**

It may be possible to produce a configurable and extendable toolkit supporting a very wide range of paradigms by providing a large library of components from which developers can select.

SCENARIOS WITH RICH ONTOLOGIES



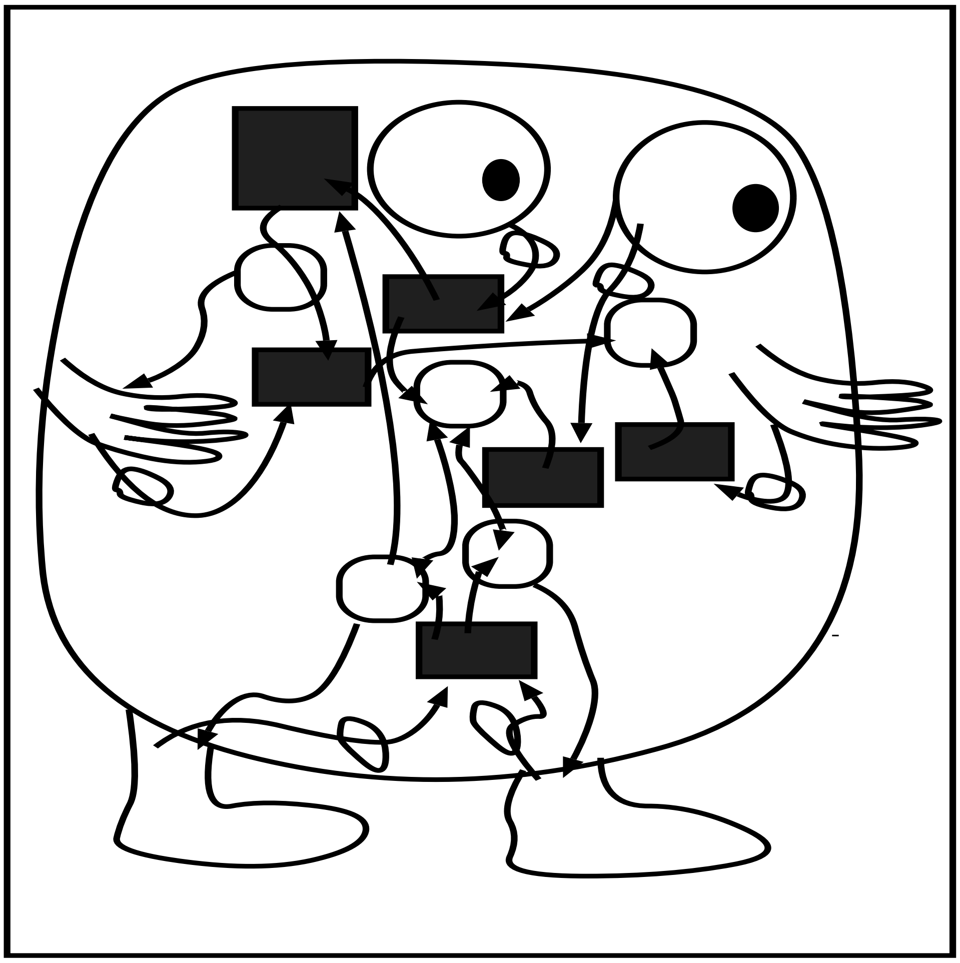
We need to cope with scenarios involving concurrently active entities, including agents which can communicate with one another, agents and objects which sense and react to other things, instruments which can act if controlled by an agent, "reactors" which don't do anything of their own accord but can react if acted on (e.g. a mouse-trap) and immobile locations of arbitrary extents and all sorts of relevant properties, including continuously varying heights and other features.

APPROACHES TO DIVERSITY

• Tools to support this diversity cannot be expected to anticipate all types of entities, causal and non-causal relationships, states, processes, etc. which can occur.
• So users should be able to extend the ontology as needed.
• One approach uses axioms defining different classes and subclasses.
• Another allows architectures to be assembled diagrammatically.
• Another approach is the use of object oriented programming, especially with multiple-inheritance.
Which is more useful is likely to depend on other factors than the nature of the ontology — e.g. how well defined the scenario is at the start.

E.g. our SIM_AGENT toolkit uses an object-oriented approach:
• Default classes are defined with associated methods.
• Users can define new subclasses, and extend or replace the methods.
• There is no fixed architecture: many different kinds of architectures can be assembled built out of interacting concurrently active condition-action rulesets.

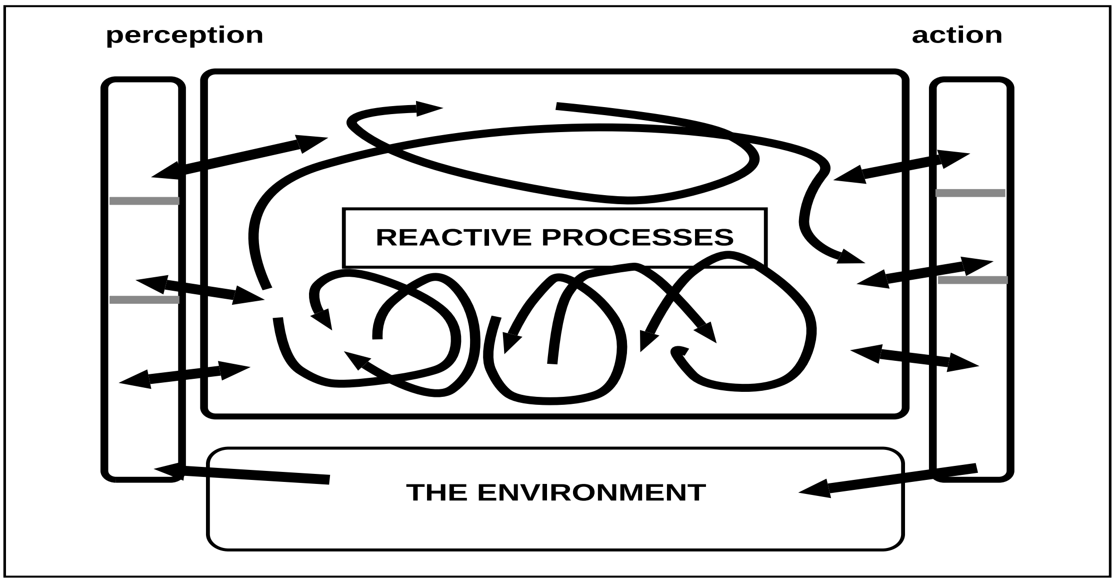
WHAT SHOULD BE INSIDE ONE AGENT?



Rectangles represent short or long term databases, ovals represent processing units and arrows represent data flow.

The toolkit should support agents with various sensors and motors connected to a variety of internal processing modules and internal short term and long term databases, all performing various sub-tasks concurrently, with information flowing in all directions simultaneously.
That still allows MANY **variants.**

REACTIVE AGENTS
HOW TO DESIGN AN INSECT?

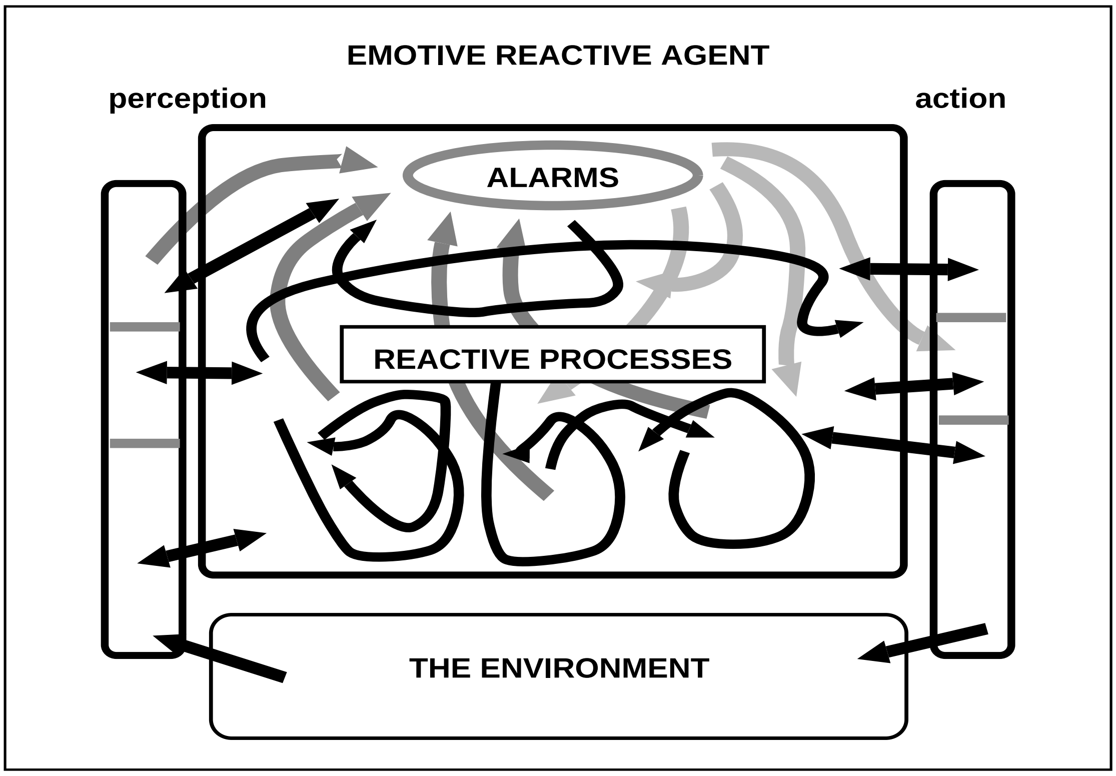


IN A REACTIVE AGENT:

- **Mechanisms and space are dedicated to specific tasks**
- **There is no construction of new plans or structural descriptions**
- **There is no explicit evaluation of alternative structures**
- **Conflicts may be handled by vector addition, simple rules or winner-takes-all nets.**
- **Parallelism and dedicated hardware give speed**
- **Many processes may be analog (continuous)**
- **Some learning is possible: e.g. tunable control loops, change of weights by reinforcement learning**
- **The agent can survive even if it has only genetically determined behaviours**
- **Cannot cope if environment requires new plan structures.**
- **Compensate by having large numbers of expendable agents?**

NB: DIFFERENT PROCESSING LAYERS CAN BE SUPPORTED: E.G. HIGH ORDER CONTROL LOOPS.

EMOTIVE REACTIVE AGENTS



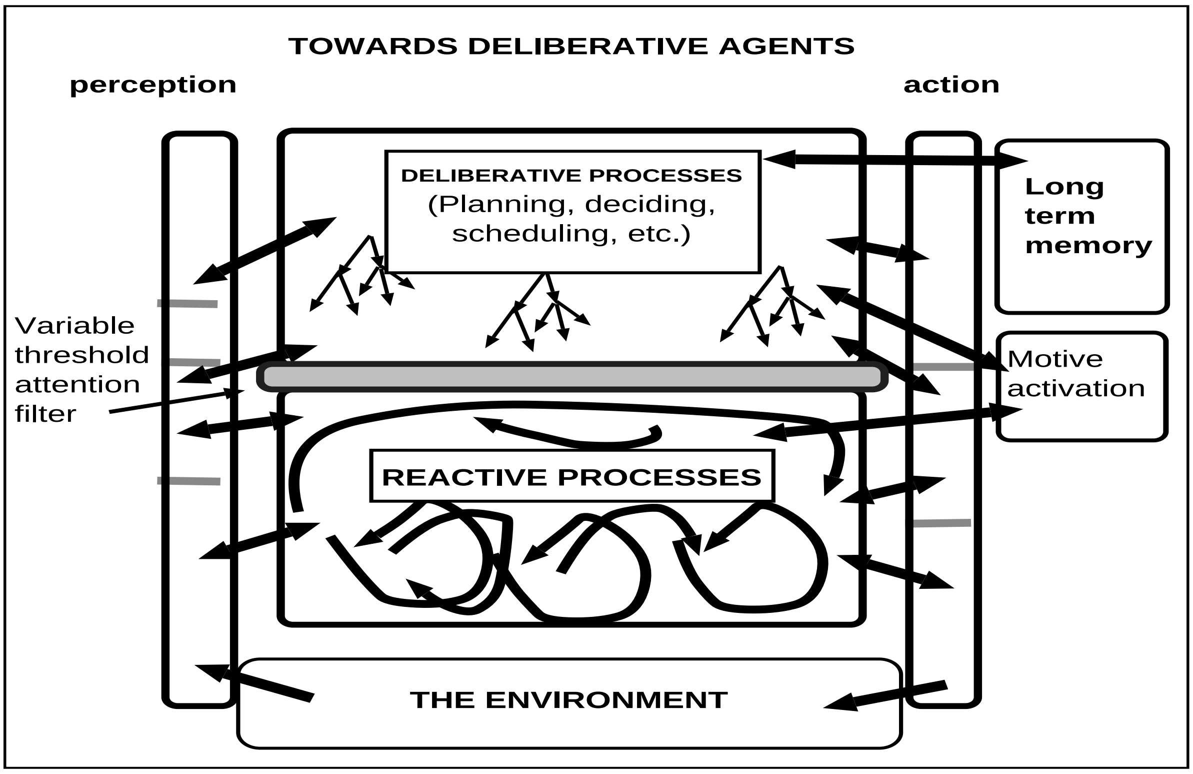
Some sort of "override" mechanism seems to be needed for certain contexts

AN ALARM MECHANISM:

- **Allows rapid redirection of the whole system**
- **sudden dangers**
- **sudden opportunities**
- FREEZING
- FIGHTING
- FEEDING
- ATTENDING (VIGILANCE)
- FLEEING
- MATING
- MORE SPECIFIC TRAINED AND INNATE AUTOMATIC RESPONSES

Damasio and Picard call these "Primary Emotions"

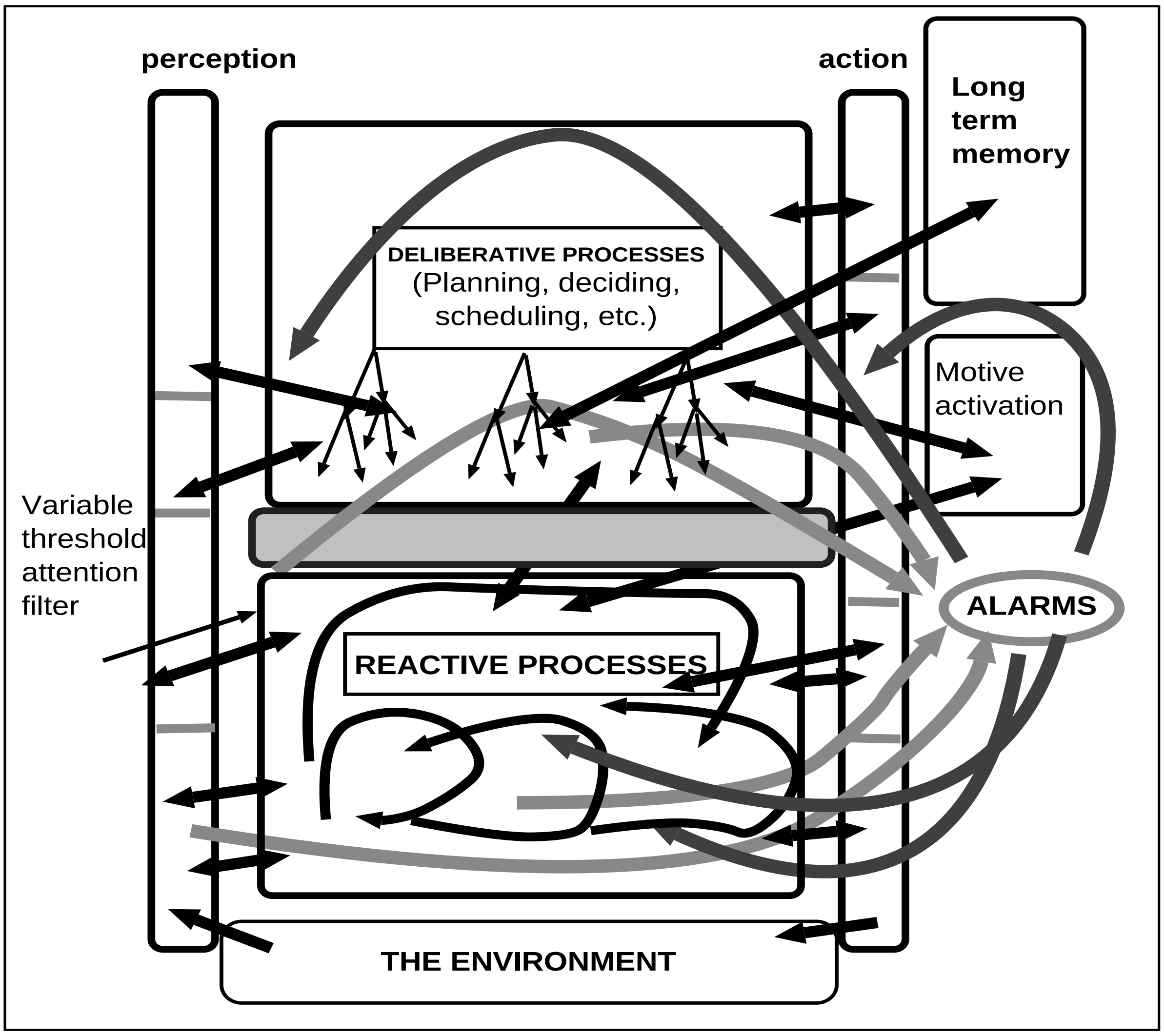
REACTIVE AND DELIBERATIVE LAYERS



IN A DELIBERATIVE MECHANISM:

- **Motives are explicit and plans are created**
- **New options are constructed and evaluated**
- **Mechanisms and space are reused serially**
- **Learnt skills can be transferred to the reactive layer**
- **Sensory and action mechanisms may produce or accept more abstract descriptions (hence more layers)**
- **Parallelism is much reduced (for various reasons):**
  - LEARNING REQUIRES LIMITED COMPLEXITY
  - SERIAL ACCESS TO (PARALLEL) ASSOCIATIVE MEMORY
  - INTEGRATED CONTROL
- **A fast-changing environment can cause too many interrupts, frequent re-directions.**
- **Filtering via dynamically varying thresholds helps but does not solve all problems.**

REACTIVE AND DELIBERATIVE LAYERS WITH ALARMS


perception
action
Long term memory
DELIBERATIVE PROCESSES (Planning, deciding, scheduling, etc.)
Motive activation
Variable threshold attention filter
ALARMS
REACTIVE PROCESSES
THE ENVIRONMENT

AN ALARM MECHANISM (The limbic system?):
Allows rapid redirection of the whole system

- **Freezing in fear**
- **Fleeing**
- **Attacking (to eat, to scare off)**
- **Sudden alertness ("what was that?")**
- **General arousal (speeding up processing?)**
- **Rapid redirection of deliberative processes.**
- **Specialised learnt responses**

Damasio: cognitive processes trigger "secondary emotions".

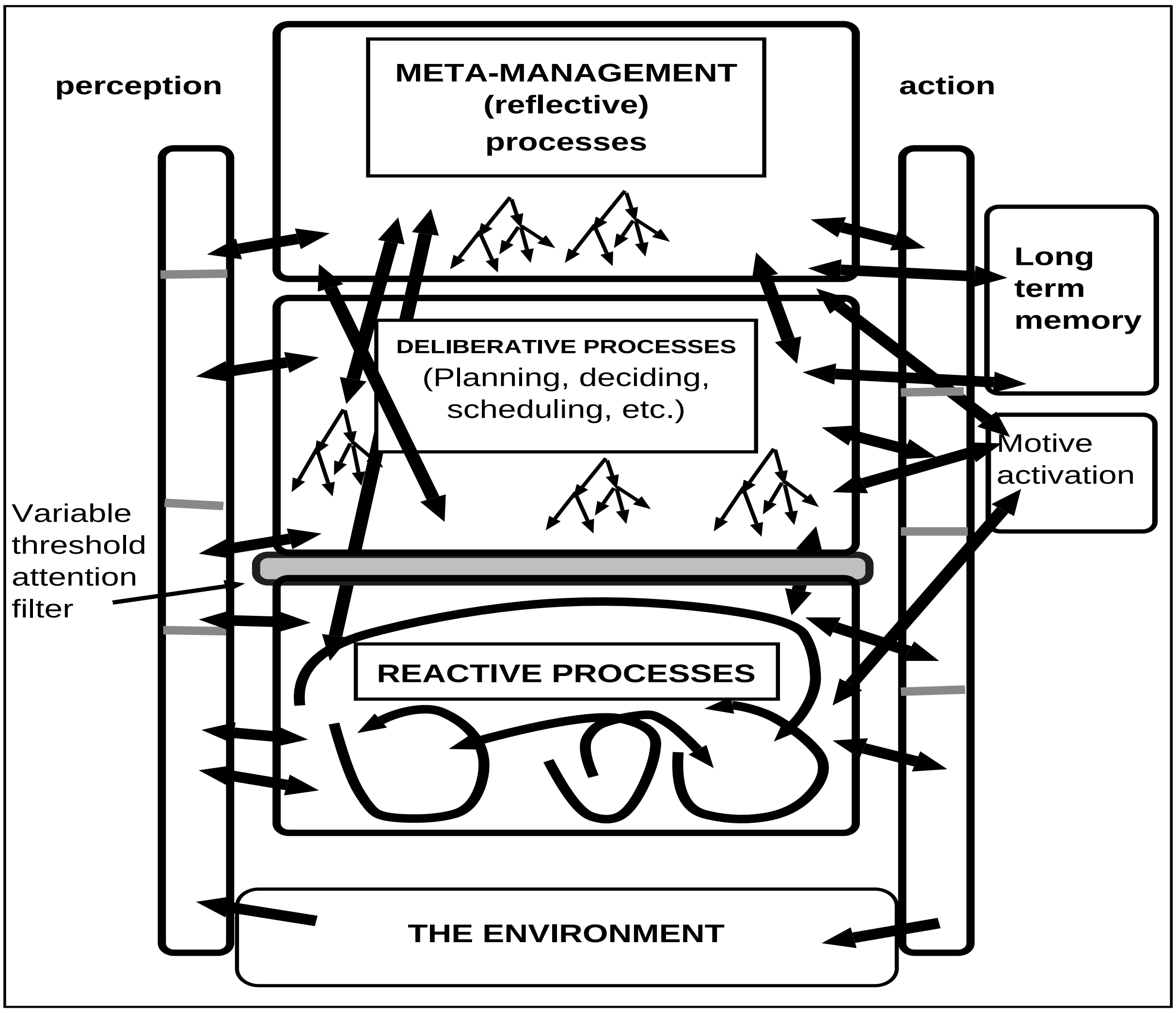
SELF-MONITORING (META-MANAGEMENT)

Deliberative mechanisms with evolutionarily determined strategies may be too rigid.

Internal monitoring mechanisms may help to overcome this if they

- **Improve the allocation of scarce deliberative resources**
  e.g. detecting "busy" states and raising interrupt threshold
- **Record events, problems, decisions taken by the deliberative mechanism,**
- **Detect management patterns, such as that certain deliberative strategies work well only in certain conditions,**
- **Allow exploration of new internal strategies, concepts, evaluation procedures, allowing discovery of new features, generalisations, categorisations,**
- **Allow diagnosis of injuries, illness and other problems by describing internal symptoms to experts,**
- **Evaluate high level strategies, relative to high level long term generic objectives, or standards.**
- **Communicate more effectively with others, e.g. by using viewpoint-centred appearances to help direct attention, or using drawings to communicate about how things look.**

AUTONOMOUS REFLECTIVE AGENTS


perception
META-MANAGEMENT (reflective) processes
action
Long term memory
DELIBERATIVE PROCESSES
(Planning, deciding, scheduling, etc.)
Motive activation
Variable threshold attention filter
REACTIVE PROCESSES
THE ENVIRONMENT

META-MANAGEMENT ALLOWS

- **Self monitoring (of many internal processes)**
- **Self evaluation**
- **Self modification (self-control)**

NB: ALL MAY BE IMPERFECT

- **You don't have full access to your inner states and processes**
- **Your self-evaluations may be ill-judged**
- **Your control may be partial (why?)**

“META-MANAGEMENT” PROCESSES MIGHT:

- **Promote various kinds of learning and development**
- **Reduce frequency of failure in tasks**
- **Not allow one goal to interfere with other goals**
- **Prevent wasting time on problems that turn out not to be solvable**
- **Reject a slow and resource-consuming strategy if a faster or more elegant one is available**
- **Detect possibilities for structure sharing among actions.**
- **Allow more subtle cultural influences on behaviour**

ALARM MECHANISM CAN BE EXTENDED

- **Inputs from all parts of the system**
- **Outputs to all parts of the system**
- **Fast (stupid) reactions driven by pattern recognition**

(Too complex to add to diagram: imagine an octopus on one side with tentacles extending into all the other sub-mechanisms, getting information and sending out global control signals. Humans seem able to learn to suppress some of these global signals. We can also learn to generate some of them voluntarily, e.g. in certain kinds of acting.)

NOTE: In humans there’s also a very complex chemical infrastructure with multiple subtle forms of long term and short term control (e.g. affecting mood, arousal, etc.).

Against Damasio and Picard: THERE COULD BE EMOTIONS AT A PURELY COGNITIVE LEVEL – AN ALARM MECHANISM INTERRUPTING AND DIVERTING PROCESSING WITHOUT GOING THROUGH THE PRIMARY EMOTION SYSTEM

SOME REQUIREMENTS

Sub-components of such agents may act more or less concurrently and asynchronously, e.g.

- TAKING IN NEW PERCEPTUAL INFORMATION,
- PROCESSING NEW COMMUNICATIONS,
- GENERATING NEW MOTIVES,
- COMPARING MOTIVES TO DECIDE WHICH TO ADOPT AS INTENTIONS,
- FORMULATING PLANS TO ACHIEVE INTENTIONS,
- DECIDING WHETHER TO REVISE INTENTIONS,
- EXECUTING PLANS, PERFORMING ACTIONS,
- MONITORING ACTION PERFORMANCE,
- DECIDING WHETHER TO REVISE PLANS OR STRATEGIES,
- REVISING THEM!
- GENERATING OR INTERPRETING LINGUISTIC COMMUNICATIONS,
- LEARNING OF MANY KINDS,
- MONITORING AND EVALUATING INTERNAL PROCESSES (E.G. PROBLEM SOLVING PROCESSES OR ATTENTION SWITCHING PROCESSES),
- REVISING STRATEGIES, ETC.,
- INTERRUPTING DELIBERATION TO ATTEND TO NEW INFORMATION OR NEW MOTIVES, ETC.,
- PERFORMING ROUTINE TASKS NON-ATTENTIVELY, ETC. ETC.)

It would be nice to handle continuous motion, but...

INTERNAL COMPLEXITY

HIERARCHIC CONCURRENCY AND SPEED CONTROL

The tools must also support not only agents which act concurrently and asynchronously, but also components within individual agents which act concurrently and asynchronously, and components within components...

I.e. Discrete event simulation systems must support a hierarchical structure.

We need to be able to control relative speeds of different components (e.g. to explore resource-allocation strategies and architectures for dealing with problems due to resource limits, e.g. filters with interrupt thresholds, meta-management).

COMBINING METHODOLOGIES

Different *types* of mechanisms are likely to be required, including rule-based reactive systems, neural nets, parsers, meaning generators, sentence generators, pattern-directed associative knowledge stores, low level image analysers mainly crunching numbers, high level perceptual mechanisms mainly manipulating structures, simulations of other agents, event-driven and interrupt-driven modules etc.

This in turn imposes a requirement for using different kinds of programming language or specification language for different subtasks.

It should be possible to have different sorts of agents with different architectures geared to different tasks and requirements.

LEARNING AND SELF MODIFICATION

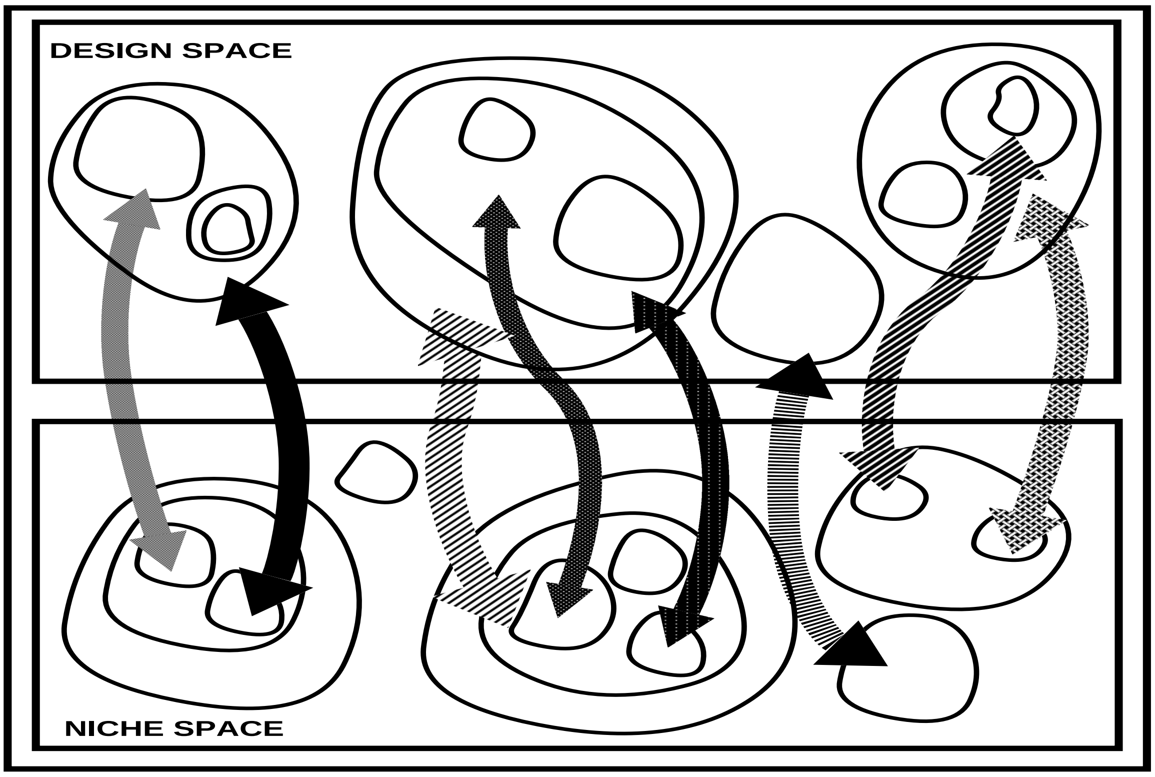
INCREASING ARCHITECTURAL COMPLEXITY INCREASES SCOPE FOR LEARNING AND DEVELOPMENT WITHIN AN INDIVIDUAL

● The more components there are, the more things there are that might be improved either by being self-adapting or via external mechanisms.
● The more components there are, the more scope there is for new links to be added, or for links to be modified (e.g. carrying richer messages).
● The more sophisticated the agent the more scope there is for improvements based on developing new representations.

ARCHITECTURAL CHANGE
Individual agents, through learning or development, may need to be able to modify *their own* architectures, either to simulate biological processes of growth and development, or because applications of artificial agents require changes of competence at run time (e.g. agents extending themselves with new "plug-in" components at any level).

THERE ARE MANY TRADEOFFS
● Between having agents “born” competent vs having them learn for themselves
● Between improvements through individual learning and development and improvements through social developments.
● Between having large numbers of simple (and expendable) agents and having small numbers of larger and more sophisticated agents

WE NEED TO EXPLORE MAPPINGS BETWEEN DESIGN SPACE AND NICHE SPACE
Between different designs and different sets of requirements



Arrows linking designs and niches depict different sorts of complex "fitness" relationships (usually involving tradeoffs). Changes in one design can alter the niche of another, which in turn can lead to design changes, which alter the niche of the first. Interacting trajectories in both spaces may involve multiple feedback loops.

DIFFERENT KINDS OF TRAJECTORIES (some discontinuous):

i-trajectories POSSIBLE FOR AN INDIVIDUAL

e-trajectories POSSIBLE ACROSS GENERATIONS

r-trajectories POSSIBLE FOR AN "EXTERNAL" REPAIRER

CHALLENGES FOR THEORISTS

● It seems likely that the sort of complexity outlined above will be required even in some safety critical systems. Can we possibly hope to understand such complex systems well enough to trust them?

● Will we ever be able to automate the checking of important features of such designs?

● The design of systems of such complexity poses a formidable challenge. Can it be automated to any useful extent?

● Do we yet have good languages for expressing the *requirements* for such systems (e.g. what does "coherent integration" mean? What does "adaptive learning" mean in connection with a multi-functional system?)

● Do we have languages adequate for describing *designs* for such systems at a high enough level of abstraction for us to be able understand them (as opposed to millions of lines of low level detail)?

● Will we ever understand the workings of systems of such complexity?

● How should we teach our students to think about such things?