

THE COMPUTER REVOLUTION IN PHILOSOPHY (1978)

Aaron Sloman

[Book contents page](#)

This chapter is also available in [PDF format here](#).

PART TWO: MECHANISMS

CHAPTER 6

SKETCH OF AN INTELLIGENT MECHANISM[Note 1]

6.1. Introduction

Much of this book is concerned with describing aspects of the human mind. In the present chapter, I shall try to provide an overall theoretical framework by describing very briefly and crudely a type of mechanism which could be represented on a computing system and which would simulate some of the important general features of the human mind mentioned in other chapters. Such a computer model would provide at least a tentative and partial explanation of certain forms of human possibilities, including the possibility of accidental discoveries, and creative redeployment of old resources.

In particular, I want to undermine a common misconception about computers, namely that however complex the programs that run in them they are always essentially unintelligent, uncreative mechanisms, blindly following simple rules one at a time. Such a description may well be true of the underlying electronic components, just as it may well be true to say that a human brain is always essentially an unintelligent uncreative bundle of nerve-cells (or an assemblage of atoms) blindly reacting to one another in accordance with chemical and physical laws of nature. But just as the latter description may omit some important features of what a brain can do, so also the former description omits important 'high-level' features of complex computer programs. What is true of a computer need not be true of a program, just as what is true of a brain need not be true of a mind. In both cases the whole is far more than the sum of its parts.

I am not trying to explain phenomena which are unusual, hard to observe, and known only to experimental psychologists. The facts about people that I take for granted and attempt to account for are facts which we all know, though we may not all reflect on them, they are part of our shared common-sense.

6.2. The need for flexibility and creativity

In particular, I shall try to sketch the overall architecture of a computing system which could cope with a variety of domains of knowledge in a flexible and creative way, so that, like people, it can use available information, skills and procedures in order to solve new problems, or take decisions in new situations, in ways which were not explicitly foreseen or planned for by the programmer. The architecture to be described is not physical but computational. It concerns global organisation, rather than detailed mechanisms, and the sub-mechanisms are virtual machines rather than physical machines. (Though they are all implemented in physical machines, e.g. brains.)

Many notable examples of creativity are discussed in A. Koestler's *The Act of Creation*. However, we can also observe frequent examples of what seems to be essentially the same kind of flexibility and creativity in the daily life of ordinary persons, in our efforts to cope with spilt milk, ungrammatical sentences, unfamiliar typewriters, blind alleys, broken suspenders, lost keys, illegible handwriting, mixed metaphors, puzzle pictures and veiled insults. The child who uses his old counting ability as a basis for answering new questions (like 'what number comes before five?') is as creative as any artist or scientist. How can we explain this flexibility and creativity?

What is required is a design for a computing system which is able to cope with types of possibility not covered by the programmer's analysis. More precisely, it is necessary to combine into a single system, competence in a variety of domains, in such a way that expertise in two or more domains can be combined creatively and flexibly in dealing with novel situations or problems. Instead of the programmer doing the analysis of all types of possibilities in advance, the program should be able, in at least some cases, to do the analysis when it is appropriate to do so, and to record the results for future use.

6.3. The role of conceptual analysis

Some insight into the mechanisms underlying human flexibility can be found in a philosophical analysis of such familiar concepts as *notice, alert, interested, puzzled, surprised, understand, cautious, attend, careless, reckless, discern, try, recognize that*, and many more. This analysis shows that to explain, by means of a computer simulation, how it is possible for available resources to be deployed in an intelligent and creative way, we need at least to construct a system which can act on the basis of multiple purposes or motives. Moreover, in the course of executing some action it must be able:

- a) To notice something it was not explicitly looking for
- b) To interrupt, abandon, modify or suspend some or all of the current action,
- c) To search through its stock of resources for items which satisfy a current requirement or need, possibly in an unforeseen way (see (a)),
- d) To relate parts and effects of one action to purposes preferences or other motives *besides those which generated the action*, or more generally, to relate facts to problems and purposes not currently being attended to.

These abilities require the system to contain mechanisms which facilitate communication of information between different sub-processes in an unplanned way: the programmer need not have anticipated each possibility inherent in the system. I shall now give a sketchy description of how this might be achieved. Several steps in the construction of such a system have already been taken by people designing artificial intelligence programs (e.g. Sussman, 1975).

Note: 2004 Sussman's book (based on his PhD thesis) seems to be available here.
<http://portal.acm.org/citation.cfm?id=540310&dl=ACM&coll=portal>

6.4. Components of an intelligent system

I shall describe (at a very high level of abstraction) some of the *structures* the system will have to contain; some of the *procedures* (or programs) that will be needed to inspect, construct, manipulate, and use those structures, and some of the *processes* that will be generated when those procedures are executed. The structures and processes may be either inside the mechanism or in the environment.

However, it will be seen to be useful to blur the distinction between the mind of the mechanism and the environment. (This blurring in one form or another has a long philosophical history. See, for example, Popper's '*Epistemology without a knowing subject*', reprinted in his *Objective Knowledge*. As he points out, Plato, Hegel and Frege had similar ideas.)

We shall discuss interactions between the following structures:

1. an environment,
2. a store of factual beliefs and knowledge,
3. a store of resources (for instance a dictionary and previously learnt procedures for making things, solving problems, etc.),
4. a catalogue of resources,
5. a motivational store,
6. a process-purpose index (action-motive index),
7. various temporary structures associated with ongoing information processing.

Many more or less temporary internal and external processes (actions) will be generated by these structures. There will also be the following more permanent processes ensuring that the actions which occur are relevant to the current motives and that intelligent use is made of previous knowledge and new information:

1. central administrative processes, not to be confused with an homunculus (see also p. 244, Chapter 10, below);
2. a set of monitoring processes, including both permanent general-purpose monitors and others which are more specialised and are set up temporarily in accordance with current needs;
3. a retrospective analysis process, reviewing current beliefs, procedures and plans on the basis of records of previous occurrences.

The system must have several kinds of processes running simultaneously, so that implementing it on a computer will require multi-processing time-sharing facilities already available on many computers, and used in the POPEYE vision program described later in chapter 9. This *global* parallelism is an important requirement for our mechanisms, though concurrent processes can be implemented in a very fast serial machine.

6.5. Computational mechanisms need not be hierarchic

The main parts of the mechanism will be described separately in terms of their functions. However, computing models, unlike previous kinds of mechanisms, should not be thought of as composed of several interlocking parts which could exist separately, like parts of an engine or human body. Normal concepts of part and whole do not apply to computing structures and programs.

For instance, two data-structures stored in the memory of a computer, containing pointers to their elements, may contain pointers to each other, so that each is an element of the other. This can be illustrated by so-called 'list-structures'.

Thus, a list A may contain, among other things, the list B, while list B contains the list A. A is then an element of B and B an element of A (which is not possible for physical collections). A list may even be an element, or part, of itself. Examples of circular structures will be found below in chapter 8. (For further details consult a manual on some list-processing programming language, e.g. Burstall, *et al.* 1973, or Foster, 1967, or a manual on Lisp, Prolog, Scheme, or Pop-11).

Similarly, computer programs may be given names, in such a way that at a certain point in the set of instructions defining one program. A, there is an instruction of the form 'If condition X is satisfied then run program B', while program B contains a similar call of program A. Program A may even contain an instruction to run itself. (These are examples of 'recursion'.) Such circular programs are able to work provided certain conditions are satisfied which can be roughly summed up by saying that during execution the series of nested, or embedded, calls to programs (sub-routines) must eventually produce a case where a particular program can run without meeting the conditions which make it call another: it can then do its job and feed the result back to the program which called it, which can then get on with its job, and so on. This is commonplace in programming languages which permit recursion, such as ALGOL, ALGOL68, LISP, or POP-2.)

In such cases, we can say that program A is part of program B, but B is also part of A. More complex chains or networks of such circular relationships between programs are possible. Similarly, human abilities, as we shall see, combine to form complex systems in which normal hierarchic part-whole relationships are violated: each of two abilities may be a part of the other. For instance, the ability to read someone's hand-writing may be a part of the ability to understand his written sentences, and vice versa.

Because these ideas have been made precise and implemented in the design of computing systems, we can now, without being guilty of woolly and unpackable metaphors, say things like: the environment is part of the mechanism (or its mind), and the mechanism is *simultaneously* part of (i.e. 'in') the environment!

We turn now to a sketch of structures, programs and processes in a mechanism to simulate purposiveness, flexibility and creativity. I cannot give more than a bird's eye view of the system at present. My description is deficient, in that it does not provide a basis for a team of experienced programmers to construct such a system. At best, it provides a framework for further research.

6.6. *The structures*

A structure is a complex whole with parts standing in various kinds of relationships. The chapter on numbers (Chapter 8) gives several examples. As parts are removed, replaced, or added, or relationships between parts changed, processes occur within the structure. In order that a structure be usable by the system, there must be mechanisms (already to be found in many computing systems) which are able to identify the parts, read off or compute their relationships, match one structure against another, interpret one structure as representing another, and perhaps perform deductions in order to extract implicit information about the contents of the structure. The system may also be able to modify or 'update' the structure, by modifying the components or their properties and relationships, or adding new components. All the structures listed below will be capable of changing, but some will be much more dynamic than others. Such manipulable structures are often referred to as 'data-structures'. They function as complex symbols, or representations.

The different structures about to be mentioned are listed separately in terms of their different functions in the system. But they need not exist separately. As already remarked, one structure may be part of another which is part of it. Some of the structures are in the mind (or computer), some not.

6.6.(a) *The environment*

This is a domain in which configurations can exist and processes occur, some but not all of them produced by the mechanism itself, and some, but not necessarily all of them, perceivable by it. For instance, the environment will normally be a space-time domain inhabited by the mechanism.

But for human beings it may also include, at the same time, a more abstract culturally determined domain, such as a kinship system, or a system of socio-economic relationships, within which the individual has a location. Some of the 'innards' of the mechanism or person may also be thought of as part of the environment, since the system can examine and act on them! (See chapter 10 for more on this.) Similarly, parts of the environment, like internal structures, may be used as an information store (blazing a trail, writing a diary, 'reading' the weather-signs, putting up signposts), so that the environment is part of the store of knowledge, that is, part of the mind.

6.6.(b) *A store of factual information (beliefs and knowledge)*

This is a set of descriptions or representations of aspects of the form and contents of the environment (including descriptions of some of the system's own 'innards'). It may include specifications of the current situation, previous history, (especially records of the system's own actions and their effects), and predictions or expectations about the future.

Several different kinds of language or symbolism may be used, for instance sentences, networks representing sets of relationships, maps, diagrams, and templates. Some of the information may be procedural, for example, in the form of routes and recipes. The information will necessarily be incomplete, and may contain errors and inaccuracies. There may even be undetected inconsistencies, and mistakes about the system's *own* states and processes. We do not necessarily know our own minds.

What gets into the store will depend not only on what stimuli reach the sense organs, but also on what languages and symbolisms are available for expressing information, and on what kinds of perceptual analysis and recognition procedures (i.e. the monitors mentioned below) are available and active. (What is already in the store will also make a difference. Where things are stored will depend on indexing procedures used.)

In order that its contents be readily accessible, this store of beliefs will have to have an index or catalogue associated with it, possibly including general specifications of the *kinds* of information so far available or unavailable. For instance, it should be possible to tell that certain types of information are not present without exhaustive searches. (How long does it take you to decide whether you know what Hitler ate at his last meal?) The index may be implicit in the organisation of the store itself, like the bibliographies in books in a library, and unlike a library catalogue which is kept separate from the books. If books contained bibliographies which referred directly to locations in the library (e.g. using some internationally agreed system for shelf-numbers) the analogy would be even stronger.

6.6.(c) *A motivational store*

In a mind there will be at any time many current purposes and sub-purposes, preference criteria, constraints on permissible actions, plans for current and future actions, specifications of situations to be avoided, etc. These likes, dislikes, preferences, principles, policies, desires, hopes, fears, tastes, revulsions, goals, ambitions, ideals, plans and so on, have to be accessible to the system as a basis for decision-making or execution, so they will need to be formulated, in an appropriate symbolism, in a motivational store.

If they are not explicit, but are implicit in decision-making procedures, then it will be much harder for the system to become aware of the reasons for what it does, and to revise its decision-making strategies. (Compare the discussion of consciousness in chapter 10, below. A more detailed analysis would distinguish first-order motivators, such as goals or desires, from second-order motive-generators or motive comparators, e.g. attitudes, policies and preferences.)

Some of the contents of the store will have been generated on the basis of others, for instance as means, plans, or strategies for achieving some end. (This store must not be thought of as a 'goal-stack' with only the last added goal accessible at any one time as in some over-simple computer models.)

The representational devices may be varied: for instance some *motivational* information might be stored in an apparently *factual* form within the previously mentioned store of beliefs, for example, in sentences like 'Jumping off objects is dangerous', or 'Nasty people live in town T'. This can work only so long as adequate procedures are available in at least some contexts for finding and using this sort of information when it is relevant to deciding what to do.

The processes produced by the mechanism, that is its actions, whether internal or external, will be generated, modified, controlled, interrupted, or terminated, by reference to the contents of the motivational store, in ways to be explained briefly below. Such purposive actions may include planning processes, the construction of new motives, problem-solving processes, external movements, manipulations of objects in the environment, processes of modifying plans or actions generated by other processes, and also perceptual or monitoring processes.

One of the constraints on the design of a human-like intelligent system is the need to act with speed in many situations. This has some profound design implications. In order that rapid decisions may be taken in a complex world there will have to be a very large set of 'rules of thumb', including rules for deciding which rule to use, and rules for resolving conflicts. This is almost certainly incompatible with assumptions made by economists and some moral philosophers about how (rational) people take decisions. For instance, there need not be any overall tendency for the rules to optimize some abstraction called 'utility'.

At any time, some of the purposes or other motivational factors may not yet have generated any process of planning or action: for instance, a purpose may have been very recently generated as a new sub-purpose of some other purpose, or it may have a low priority, or there may not yet have been any opportunity to do anything about it, or it may be a conditional purpose (do X if Y occurs) whose condition has not been realised, or some other purpose or principle (for example, a moral principle) may override it. *Thus many existing motivational factors may generate no decisions.*

Similarly, plans and decisions that have been formulated on the basis of motives may still not have generated any action for analogous reasons.

6.6.(d) A store of resources for action

This includes not only usable objects in the environment, such as tools, materials, sources of information, teachers, and so on, but also the store of factual beliefs, and, most importantly, a set of readily available programs, or procedure-specifications, some of which may use one another recursively (as explained above in section 6.5, p. 116).

Among the resources should be linguistic or symbolic abilities. Those are needed for formulating problems, purposes, procedures and factual information. Chapter 2 indicates some of the reasons why notational resources can be very important. Chapter 7, below, explains why different kinds of symbolisms may be required for different sorts of tasks or sub-tasks. Other resources would include

procedures for constructing plans or routes (for example, with the aid of maps), procedures for getting information and solving problems, such as problems about why an action went wrong, and procedures for constructing, testing and modifying other procedures. (See Sussman 1975 for a simple working example.)

That is to say, the resources store will include collections of 'intelligent' programs of the sorts currently being produced by workers in artificial intelligence. The concept of a resources store, like the concept of an environment, expands to swallow almost everything! This is why a catalogue is necessary.

6.6.(e) *A resources catalogue*

It is not enough for resources, like objects and abilities, to be available. In order that they be intelligently usable, the system must have information about them, such as what kinds of purposes or functions they are typically useful for, the conditions under which they are applicable, likely side-effects, possible dangers, and any other information which may be relevant to selecting a particular resource and embedding it in a plan, perhaps with modifications to suit current needs and conditions.

It must be possible for new information about typical causes and effects, or requirements, of old resources to be added to the catalogue. The system may have to use pointers in the catalogue in two directions, namely, starting with some purpose or need, it should be able to use the catalogue to get at available resources which might meet that need. So pointers are needed from purpose-specifications to resources. However pointers are also needed the other way, since in selecting one resource it may often be important to know what sorts of uses, and effects, it can have besides the one which led to its selection: if some other typical use of the resource matches another current motive or need, then the resource may be 'a stone that kills two birds'. Alternatively, if a resource selected as a possible means to one end has a typical effect which will frustrate some other current purpose (or principle, or preference, etc.), then an alternative resource should be sought, or the other purpose abandoned. Those are some of the design implications that follow from the need to cope with multiple motives.

Sometimes information about typical uses and side effects of a procedure (or other resource) can be got by inspecting its *structure*. But often such things are learnt only by experience of using the resource and in the latter case we need *explicit* additional entries.

For a very large store of resources, as in the human mind, the catalogue will have to be highly structured, for instance in the form of a tree, with lower levels giving more details than higher levels. The organisation of the catalogue may be partly implicit in the searching and matching procedures. As indexing can never be perfect, the system will have typically human failings, no matter how fast and large a computer is available. (This is contrary to some optimistic pronouncements about the way bigger and faster computing systems will enable super intelligences to be made.)

This catalogue of resources, like the index to factual beliefs, need not be physically separate from the store of resources: it may be partly implicit in the organisation of the store.

6.6.(f) *A process-purpose index (or action-motive index)*

The central administrative mechanism (described below) will set various processes (internal and external actions) going, on the basis of analysis of contents of the current motivational base together with analysis of the resources catalogue and the store of information about the environment. Some of the processes will consist of sets of parallel or sequential sub-processes, which in turn may have a complex inner structure. Some of the processes may be lying dormant, waiting for starting signals, or

resumption signals from monitors (see below). This is commonplace in computer operating systems.

In order to be able intelligently to modify ongoing processes, terminate them, interrupt or suspend them, change their order, and so on, in the light of new information, the system will have to have information about which processes and sub-processes are generated by any given motive, and which motives lay behind the initiation of any one process.

The function of a process-purpose index is to store this information about the reasons for various actions. It may need to be modified whenever a new process is initiated or an old one terminated, or when any of the reasons for doing something change, for example, if one of the birds a stone was intended to kill turns out to be already dead. The system will thus have access to the reasons why it is doing things. Faults in the procedures for keeping the process-purpose index up to date may account for some pathological states.

So if a process generated by purpose P1 accidentally achieves a purpose P2, and this is detected by the monitors, then the index shows which other processes were generated by P2, and can therefore be terminated, unless the index still contains a pointer from one of them to some other as yet unfulfilled purpose P3. Other uses of the process-purpose index will be mentioned below.

Perhaps one of the most important reasons why it is necessary to be able to be in the midst of several different processes at once, is that this provides opportunities to *learn* from accidental interactions between processes. The process-purpose index, which relates current activities to the reasons for doing them makes it easier to achieve such learning. For example, one might learn that a certain purpose can be achieved in a new way, because of an unexpected interaction between the old strategy for achieving it, and some other activity.

The *process-purpose index* should not be confused with the relatively more static, less changeable, *resources catalogue*. Their functions are different. For instance, a particular procedure may be selected, using the resources catalogue, in order to achieve purpose P1, and then executed. While the process of execution is going on, the same procedure may be selected again in order to achieve another purpose P2. We thus have two processes (actions) running in parallel in order to achieve different purposes, yet the same procedure (or program), a relatively permanent resource, controls them.

A clear example of this is a person playing two games of chess simultaneously, and using the same strategy in the two games for at least part of the time. If one of the opponents makes a move requiring that strategy to be abandoned, the process of executing it has to be terminated in one game but *not* in the other.

The resources catalogue contains the relatively permanent information (modifiable in the light of experience) that this strategy is normally useful in such and such circumstances for achieving certain types of advantages. The process-purpose index, however, relates not the strategy itself, but, for example, two current *executions* or *uses* or *activations* of the strategy, which may have reached different stages of advancement, to two current purposes. Similarly the ability to multiply may be used twice over in evaluating the following expression:

$$\frac{(17-12) \times (6+5)}{(3 \times 2)}$$

The process-purpose index would also have an important place in planning activities, when instead of real executions of strategies the index would contain pointers to representations of possible executions of strategies.

6.6.(g) Temporary structures for current processes

At any time, various ongoing processes will have associated with them structures containing information about partial results, current values of variables, next instruction or procedure step, current subgoals, where to send results, and so on.

Once the importance and ubiquity of such structures in a complex goal-directed information processing system has been understood, the distinction sometimes made between two kinds of memory -- short-term and long-term -- evaporates, for instance, in connection with a plan carried out over a period of several years.

Note added April 2004

That point was very badly expressed, or just wrong. What I think I was trying to say in 1978 is that there are different memories with different time-scales and different functions, and assuming there are only two kinds, short-term and long-term memory, as some people appeared to claim in those days, was a mistake.

The full details of these temporary structures need not be globally accessible in the same way as some of the previous structures. That is to say, they are private to the particular processes which use them. It may be, however, that certain local computations, are automatically reported up to a global level whenever they occur, such as estimates of time or computing space, or other resources needed for a process to be completed. This might be done by monitors, described below.

Some of the temporary workspace may be outside the system, for instance a shopping list, an artist's rough sketches, an engineer's calculations. Even a half-completed object is an extension of short term memory for the constructor.

Note added April 2004

Examples would be a partially completed painting, a partial mathematical proof, a half-built shelter. A similar point was made long ago by Herbert Simon in connection with insects that produce the next step in a complex task by reacting to the current state of the environment in which they are building something. This notion is often referred to as 'stigmergy' and the phenomenon was known to entomologists in the 1950s. Good ideas are often re-discovered.

These more or less temporary structures are of no use to an intelligent system unless mechanisms are available which can bring about the sorts of processes already hinted at and elaborated below. Typical mechanisms would be procedures for accessing, using, and modifying the resources, catalogues, plans, etc. The whole system needs some kind of overall control. This is the business of a central administrative process, for which computer operating systems provide a very first (very rough) approximation.

6.6.(h) *A central administrator*

[[Paragraph added in 1986:

To some extent, parts of the system may (and will) work autonomously in parallel, e.g. posture control, control of breathing, and control of saccadic eye movements. However, since two or more needs may require incompatible actions, and since coordinating two actions rather than performing them separately may improve overall performance, it may be useful for some 'central' system to resolve conflicts and co-ordinate decisions. A 'central administrative process' may have this role.]]

The central administrative process will at various times survey the motivational base and purpose-process index and select from the unfulfilled purposes a subset for generating further planning and action. This selection may be driven partly by previously selected purposes or principles, and may use current information, such as estimates of likelihood of success or failure, knowledge about opportunities and resources available now and in the future, the current state of other actions, and so on.

Sometimes no selection can be made until a change has been made in *the set of purposes* for instance by inventing a compromise between two conflicting purposes. In at least some cases, the selection must be automatic to avoid an infinite regress of decision making.

Similarly, after certain motives or purposes have been selected for action, then in at least some cases they must invoke suitable action-generating procedures automatically, since if everything required prior deliberation or planning, nothing could ever get started. This automatic activation can happen when a current purpose closely matches a catalogued specification of a typical use of an available procedure. Monitors would be employed to reduce the risks inherent in some automatic activation.

When no matching procedure is found for a certain purpose P, in the resources catalogue, it may be possible instead to find a match for the new purpose of *making a plan for achieving P*. For instance, if the purpose 'Go to Liverpool' fails to match any current plan, then 'Make a plan for going to Liverpool' may match a typical use (that is, making plans for going places) of a procedure for constructing *routes* (for example, find an atlas containing both your current position and the destination, then . . . etc.). Again, even if one does not yet know a procedure for making *objects* of a certain type, one may have a *procedure for constructing a suitable procedure* for making those objects, by analysing specifications of a required object and available tools and materials.

In short, when first-order matching fails, second-order matching may succeed. Perhaps in some cases even higher-order matching (make a plan for making a plan for achieving P) may succeed.

Similarly, if several procedures are found to match the purpose, then a new purpose may have to be set up, namely the purpose of choosing between the available alternatives. If a choice cannot be made using the information in the resources catalogue, it may be necessary to try out some of the alternatives. (See chapter 8 for more on the difference between examining and executing procedures.) This kind of comparison of alternatives may occur at various stages in the construction of one plan, contrary to the games-theoretic analysis of human decision-making which assumes that we always choose between *complete* alternatives, without saying anything about how we construct those alternatives.

When the administrator has failed to find or produce a plan for a certain purpose, a second-order task may have to be added to the motivational base as a new unachieved purpose (i.e. finding a plan), to be attended to later if anything relevant crops up, in ways described below. (This can produce accidental

learning.) Alternatively, if the original purpose was very urgent, or there is nothing else to do at the time, then trial and error with back-tracking may be used.

Note added April 2004

Some of the above ideas later turned up in SOAR, a problem solving and learning system developed in the early 1980s. SOAR detects when an *impasse* occurs and switches to a new task, resolving the impasse. However, as far as I know, SOAR did not include the option in the previous paragraph, namely *deferring* the process of dealing with an impasse until some unspecified future date. I believe SOAR also did not include the point in the next paragraph about checking unexpected benefits and side-effects of proposed new solutions, which became an important feature of many planning systems apparently inspired by Sussman's HACKER (Sussman 1975) referred to above. For more on the ideas in SOAR see

Newell, A. (1980b). Reasoning, problem solving and decision processes: The problem space as a fundamental category. In R. Nickerson (Ed.), *Attention and Performance VIII*. Hillsdale, NJ: Erlbaum.
<http://sitemaker.umich.edu/soar>

Should a suitable procedure for achieving P be found or constructed, in any of the above ways, then analysis of its typical uses and effects (recorded in the resources catalogue), or analysis of its structure, may show that it, or a modified version, will enable more than one current purpose or motive to be fulfilled. If several suitable alternatives are available, this analysis may provide a reason for choosing between them. Or it may show that the procedure would interfere with some other current purpose. A process (action, procedure-execution) is then generated by executing the selected procedure with suitable arguments or parameters bound to its variables (for example, 'destination' might be bound to 'Liverpool' in the previous example).

The central administrator (and perhaps also some of the other currently running programs) must be able to interrupt, terminate, modify, or restart current processes (though some may be less controllable than others, for instance if they are so well-ried that possible interrupt points have been kept to a minimum). These control decisions will be taken on the basis of new information from *monitors* (described below), using the purpose-process index as described above. So the index must be changed every time a process is begun, modified, halted, or found to be capable of serving an unexpected purpose as a side effect, as well as when ongoing processes set up new sub-goals and generate corresponding sub-processes.

Some processes which include complex sets of sub-processes, may have to have their own private purpose-process indexes in their private work spaces (see p. 124), as well as being more briefly represented in the main index. They may also have their own central administrators!

Chapter 10 attempts to relate the idea of central decision-making processes to the distinction between what we are and what we are not conscious of.

6.6.(i) Perception and monitoring programs

Mechanisms must be available for inspecting the environment in which the system acts, such as familiar types of sense-organs for inspecting the external environment, and less familiar mechanisms for accessing structures within the system (the internal environment). However, all that a *physical* sense-organ can do is produce some kind of spatial or temporal array or manifold of physical values (as a television camera or microphone does). This does not yet amount to perception: it simply amounts to the production of a new structure within the system. Whether anything is thereby perceived, and what is perceived, depends on what procedures (or programs) are available for analysing the new structure, finding relationships between its parts, perhaps manipulating or modifying it (for example, correcting misprints or other errors), interpreting it, and making use of all

this either immediately or later on in the performance of actions or solving of problems.

Such perceptual procedures may involve computations of arbitrary complexity, using a great deal of background knowledge, like the perceptual procedures involved in a medical diagnosis or the tuning of a car engine. Even ordinary perception of simple shapes and familiar physical objects can be shown to presuppose considerable factual and procedural knowledge. This is why perception cannot be separated from cognition. See chapter 9 for more details.

So the system needs a collection of perceptual procedures, for analysing and interpreting various kinds of structures in various kinds of contexts. The limits of these procedures together with the limits of the sense-organs and the current store of information about the environment will define what the system is capable of perceiving. Systems with the same physical sense organs may therefore have quite different perceptual abilities as we know from variations in human perception. Thus there cannot be any such thing as perceiving things 'directly' or 'as they are in themselves'. As Max Clowes once put it: ''We inhabit our data-structures''. The same must be true of intelligent machines. So the objective/subjective distinction evaporates. (Compare Boden, 1977.)

The range of types of objects, properties and relationships that human perceptual procedures are capable of coping with is enormous. So in a sensible system they will not all be applied to every possible chunk of sensory input or meaningful structure. For instance, when you last read a page of typescript you probably did not use your ability to notice that the letters on the page were in vertical columns as well as horizontal rows; and while listening to someone talking one language you know, you do not apply the analysis procedures which would enable you to recognise in his syllable-stream the sounds of words of another language you know. Did you notice the 'let' in 'letters' or 'horizon' in 'horizontal' above? If every available analytical and interpretive procedure were applied, their outputs would form an enormous information store, and the system would then have the problem of perceiving its contents in order to make use of the information.

It seems not only sensible, but also to correspond to human experience, to have only a small selection of available perceptual programs running at any time in relation to any one piece of 'perceivable' structure, such as the structures mentioned in the previous sections or those produced by sense-organs. There are serious problems in explaining how appropriate programs are selected.

The active analysis programs may be called '*monitors*'^[note 2] and it seems to be necessary to have two main kinds of monitoring general purpose and special purpose. The former involves frequent and large-scale application of relatively simple analyses and tests which have a good chance of being relevant to a wide range of purposes and circumstances. (Is anyone calling out my name? Is something on my retina moving?) The special purpose monitors may be more complex, and will be set up only when there is a specific reason to expect that they will find something or that if they find something it will be very useful in relation to current motives.

In either case the monitor need not itself complete the analysis and interpretation of new information. Instead, what it finds may act as a *cue* (or reminder, or stimulus) which will invoke (e.g. via a catalogue or index of resources) more complex object-specific or problem-specific procedures.

For instance, if the environment is a spatial domain, then a visual retina might be designed with very many relatively simple general purpose monitoring procedures 'wired into the hardware', for efficiency, instead of being expressed as programs. So the retina might be divided into many small regions, each being constantly monitored to see whether any change has occurred in some physically detectable property (brightness, colour, graininess of texture). If a change is noted, the monitor sends an interrupt signal to inform processes which may need the information. Other general purpose monitors might be constantly monitoring these monitors to see whether something which has

consistently been reporting changes stops doing so. There may be general purpose monitors not only at the interface with the physical environment, but also at several other interfaces. Perhaps every time one of the globally accessible structures (such as the motivational base or process-purpose index) is accessed or modified by any current process, a general purpose monitor will note this and send an appropriate signal or take appropriate action (such as recording the fact for future reference). In recently developed programming languages this is achieved by 'pattern-directed procedure activations'. It is also a common feature of computer operating systems, for example, to prevent unauthorised access to information.

A very useful general purpose monitor would be one on the lookout for 'I've been here before' situations: this might enable loops, infinite regresses, and unnecessarily circuitous procedures to be detected. However, the concept of 'the same state as before' admits such varied instantiations that it cannot be tested for in general by any one procedure. General tests might therefore have to be restricted to a few possibilities, like a return to the same geographical location, much more specialised monitors being required if other kinds of repetition are to be detected another source of fallibility in complex systems.

This will not work if records of previous states are not retained. Alas, people do not remember everything, not even their own actions. Repetitions often go undetected, like recounting their exploits or telling you a joke for the nth time. However, we shall see below that remembering apparently useless things may be an essential pre-requisite for certain kinds of intelligent behaviour and learning.

A 'found something' signal from a general purpose monitor may function simply as an *invitation* to some other program or monitor to look more closely, applying special purpose perceptual procedures to see if the occurrence is important to current motives or processes. Depending on what else is going on, the invitation may be ignored, or the new information may simply be stored, without further analysis, in case it will be useful later on. (Note that this presupposes some indexing procedure.)

Special purpose monitors may be much more complex, may have a much more transient existence, and may be set up at all levels of complexity in the system. For instance, in dealing with someone we know to be 'difficult' we need to be on the look-out for danger-signals in their behaviour. And while searching for a proof of some mathematical formula, one may have good reason to suppose that if certain sorts of intermediate results turn up in one's calculations they will enable an easy proof to be found, whereas if others turn up they will show that the formula was not provable after all. In that case one could set up monitors to be constantly on the lookout for the 'accidental' production of such results. (For examples, see Wertheimer *Productive Thinking*.) The tests for the occurrence of such special cases need not be at all trivial, and it may be necessary to make inferences from obscure cues, learnt in the course of considerable previous experience.

Watching out for multiplication or division by zero when simplifying equations illustrates this: zero may be heavily disguised in an expression like:

$$a^2 + (a + b)(b - a) - b^2$$

So the monitoring required will have to be pretty sophisticated. The same applies to detecting signs of irritation, dismay, incomprehension, etc. in one's spouse or pupils.

Normally the 'something found' signal from a special purpose monitor would be less likely to be ignored than signals from general purpose monitors, partly because the latter will always be crying 'wolf' and partly because the setting up of a specialised monitor will reflect the importance of its results, for current purposes.

Discoveries of the analytical and interpretative programs constituting monitors may be added (perhaps after some filtering by intermediate monitors) to the belief system (see section 6.6.(b)), forming a record of events and discoveries. At this stage a particularly important general purpose monitor should be available to try matching each addition to the belief system against currently unfulfilled purposes, or at least a 'high priority' subset of current motives, to see whether the new information satisfies or obstructs any of them. For example, the newly discovered fact or technique may be a solution to a problem you were thinking about yesterday. If it is a *general* purpose monitor it will have to use *crude* matching techniques, so some relevant relationships will be missed unless specialised monitors are set up. (Again, we see how fallibility is a necessary consequence of complexity.)

Not every piece of new information can be stored permanently. The problems of indexing, shortage of space, searching for what is relevant etc., would make this unworkable. But it may be possible to store information for a short time in case it turns out to be relevant to some process or purpose other than that which generated it. This will be most useful in the case of 'raw' data acquired for one purpose but potentially useful for others. If only the *interpretation* of such data is stored, then useful information may be lost. So besides the interpretation made for one purpose it may be useful also to store, at least temporarily, the original uninterpreted information in case it turns out to be relevant to other purposes. It must therefore be stored in a globally accessible structure.

In order to be really flexible and creative, the system will have to be able to activate specialised monitors, from time to time, which ask the following questions about new items of

information as they turn up:

- i) Does this imply that a particular current purpose has been achieved or frustrated?
- ii) Does it imply that particular current purposes are unexpectedly near to or far from being achieved?
- iii) Does it imply that a current purpose can be achieved more efficiently or quickly or at less risk or cost, or in a more enjoyable way, etc., by modifying an ongoing process or terminating it and starting with a new strategy: that is, is there a better way of doing what is currently being done? what is currently being done?
- iv) Does it imply that any current purposes are mutually incompatible?
- v) Is this worth examining more closely to see if questions like (i) to (iv) get a positive answer after specialised investigation.

Although such questions may occasionally be answered by a simple match between a current purpose and new information, at other times the full problem-solving power of the system may be needed in order to detect the relevance of a new fact, another example of the recursive, or non-hierarchic, nature of computational systems. For instance, a stored resource may not be found by a straightforward search in the resources catalogue. However, some further analysis of what is needed may solve the problem of where to search. Alternatively, it may later be found to be related to a current problem only when, by chance, it is turned up as a result of a search generated by some *other* need, and a monitor, or the central administrator, causes its relevance to the earlier purpose to be investigated. The person who is looking for both a screwdriver and eating utensils may be more likely to recognise the knife on the table as a potential screwdriver than the person who is simply looking for a screwdriver. But he must also be able to relate the structure of the knife to the function of a screwdriver.

6.6.(j) Retrospective analysis programs

For efficient and creative learning, the system will need to analyse fairly lengthy and detailed records of events. Such records will, as already pointed out, need to contain more detailed information than is *obviously* relevant to current needs, information retained in case it turns out to be useful.

For instance, examination of a series of failures over a long period of time may suggest a generalisation about what caused them, leading to a modification of some old procedures. (Of course, some people never learn from their failures, especially their failures in dealing with other people. Why not?)

Similarly, if successes are sometimes achieved unexpectedly, the system should go back and try to find out whether enough information was previously available for the bonus to have been predicted and planned for, in which case some existing planning procedures will again need to be modified.

Records of events must also be searched for refutations of previously accepted generalisations, and for new patterns suggesting deeper explanations of previously known phenomena. In some cases, retrospective analysis of difficulties in getting at relevant stored resources may show the need for reorganisation of the catalogue of resources or the index to information. Thus, all sorts of comparisons need to be constantly going on, relating new information, old information, current motives, and possible future motives.

Once again, retrospective analysis cannot be done simply by a general purpose program, if it is to be at all deep. There must be a preliminary general analysis of unsolved problems to suggest that certain particular types of questions need to be investigated, and appropriate special purpose investigation procedures invoked or constructed.

Normally many questions like 'Was my failure due to bad luck or was there something wrong with the procedure by which I worked out a strategy?' will remain unanswered. Unanswered questions can be added to the store of unfulfilled purposes, thereby enlarging the motivational base and possibly influencing the course of events later on, if for instance, one of these problems turns out accidentally to match some information generated by another purpose.

Moreover, these unsolved problems may themselves generate new processes of experimentation or exploration, for instance in order to test some tentative hypothesis about the scope of a regularity or the explanation of a surprise. Without a major driving force provided by the need to answer questions and solve problems, it is hard to see how human infants could possibly learn as much as they do in the first years of life. It is paradoxical that the words 'play' and 'toy' are often used to denote this most important of all human activities and its instruments. It is also worth noting that unless the system in some way consciously or unconsciously distinguishes errors in its own procedures from failures due to the environment, it cannot modify its procedures and learn. Thus even new-born infants and any organism that learns, must have a rudimentary concept of 'self', contrary to popular opinion.

6.7. Is such a system feasible?

It will not be easy to construct a working computer model containing these structures, procedures and processes. Many problems have to be solved which are hardly even mentioned in the ludicrously brief specification I have given. A suitable type of environment must be chosen for the initial attempts, with a rich but interconnected variety of possible structures and processes. There are many difficulties in enabling so many processes to interact. Existing computing systems used in artificial intelligence are too small. Symbolisms will have to be developed for expressing various sorts of purposes, possibilities and plans, and for formulating entries in catalogues, the purpose-process index and the belief system.

No simple and uniform notation can be expected to work for all cases: sometimes a desired object may have to be represented in terms of a function that it can fulfil, sometimes in terms of a verbal description of its structure, sometimes in terms of a procedure for constructing it, and sometimes in terms of a template or model, similar in structure to it. Usually a combination of representations will be needed.

A language which is suitable for formulating a procedure (or program) so that it can be executed efficiently need not be equally good for constructing the procedure in the first place nor for describing how that procedure works so that its uses and limitations can be understood. The system may have to use one language while a procedure is constructed and debugged, after which it is translated (that is, compiled) into some less accessible, less intelligible, less easily modified but more efficiently executed form.

Programming such a system would be an enormous task, yet it seems that existing expertise makes it all possible in principle. For instance there are complex operating systems which permit several different processes to run on a single computer, as if in parallel (because small chunks of each are run in turn), interacting with each other as they go, and this would enable several monitoring programs and administrative programs to run at the same time as programs for planning, executing actions and retrospective analysis. The POPEYE perception project, described in a later chapter, illustrates the possibility of such parallelism, in a simple form.

6.8. The role of parallelism

It may be necessary to have a computer with several different processors, each containing its own time-sharing system, each processor being devoted to some major function described above, but all having access to the same set of stored structures, but it is not obvious that this is necessary. It is often supposed that the human brain has some tremendous advantage over electronic computers because it operates in parallel, but there is no reason to suppose that having very large numbers of processors working in parallel would be an advantage if they all had the opportunity of producing complex changes in a central store: the mess might be impossible to control.

Parallel processors might be of use only for relatively simple, general purpose monitoring of the kinds already described, such as the monitoring of a retinal array for simple events, and perhaps the monitoring of stored symbols for crude and obvious matches with widely broadcast current requirements.

Since all this can in any case be simulated on a single, serial processor, the distinction between serial and parallel physical processors has not much theoretical significance for our purposes. This is not to deny that parallel processing (which can in principle occur on a serial processor) is crucial for the kinds of interactions between processes described above.

[[Note added 2001.

This point became clearer in the 1990s and beyond when AI researchers saw the importance of *architectures* for complete systems, instead of concentrating only on representations and algorithms. See my 1992 review of Penrose: A. Sloman, 'The emperor's real mind', Review of Roger Penrose's *The Emperor's new Mind: Concerning Computers Minds and the Laws of Physics*, in **Artificial Intelligence**, 56, pp. 355--396, (available at the [CogAff web site](#)]]

6.9. Representing human possibilities

All this is part of an answer to the question: How should we represent the human mind in such a way as to do justice to the enormous variety of possibilities inherent in it? What a person experiences, thinks and does over a period of time is a single process with many sub-processes, but we know that if he is a normal person then his mind contains, during that time, the potential for many other processes which might have occurred. (Compare [Ryle, 1949](#), on dispositions). A computing system of the sort just described could represent an explanation for such human possibilities. Of course, the proposed system could not account for all major human possibilities without further complex mechanisms. For instance, nothing has been said yet about how the possibility of moods and emotions is to be explained. This would involve various kinds of disturbances of the central processes. (For instance, feeling startled is sometimes a result of rapid automatic re-organisation of a collection of plans and actions in the light of a sudden awareness of new danger.)

How wide a range of possibilities existed in such a system at any time would depend on such things as how wide a range of resources were stored in it, how complete the catalogues of resources and beliefs were, and what kinds of matching mechanisms were available. These, in turn, would depend, as in the case of a human being, on how much and what kind of previous learning had occurred. A mind contains a culture. So anthropology must be part of psychology, and vice versa.

Nobody could hope to design a complete adult humanoid robot. At best, it may be possible to produce a baby mind with the ability to absorb a culture through years of interaction with others.

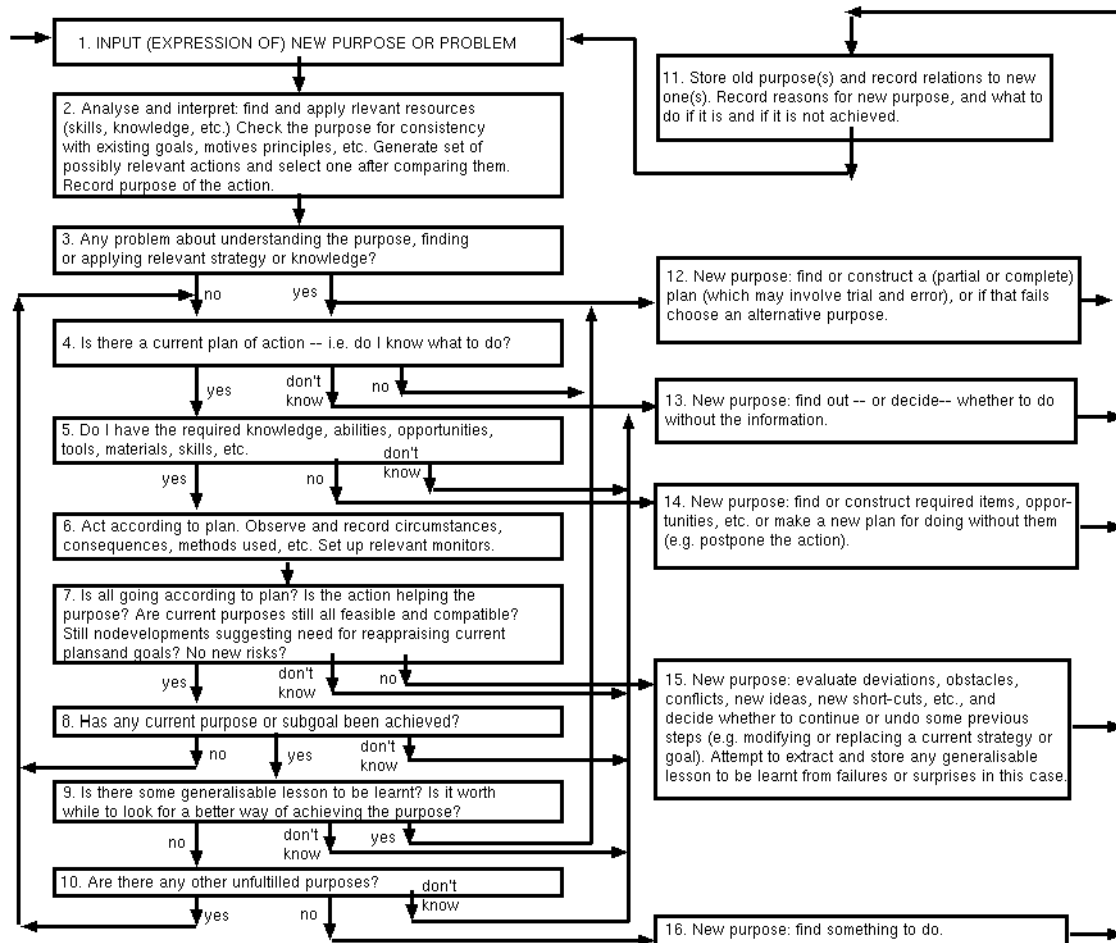
6.10. A picture of the system

It is possible to give a crude representation of some of the range of possibilities inherent in such a computing system, or a person, by means of a flow-chart. This consists of a set of boxes (nodes in a graph) representing possible states or sub-processes of the system, joined by arrows representing the transitions that can occur between them.

If four arrows lead from box A to other boxes, this implies that there are four possible states, or sub-processes, which can occur after the one represented by A. Which one does occur will, normally, depend on what sorts of things occur in phase A, which in turn may depend on features of the context, including a long history of previous processes. (The normal approach in social science and much psychology is to shirk the task of understanding such dependencies, by representing the transitions as probabilistic, and studying the frequencies with which they occur in a large sample examined superficially, instead of studying particular transitions in depth.)

Each box represents a *state* of the whole system, so a flowchart of this sort should not be confused with a chart in which the boxes represent *mechanisms* of some kind and the arrows indicate *flow* of something like energy or information. Mechanisms are not states or phases, and flow between parts is not the same thing as transition between states of the system. A flow chart is not an architecture diagram (though it may imply some architectural features.)

The chart summarising many examples of familiar kinds of human behaviour, follows. (In the original book it was on pages 138-9).



This kind of flow-chart can be misleading in various ways.

- a) Each box represents a sub-process which may have very great internal complexity, including many possible alternative sub-processes.
- b) Because the chart has loops, the same box may be entered twice, or many times but each entry will represent a different sub-process, and this is not represented in the chart.
- c) The action of monitors is not represented. They run concurrently with the processes depicted and can detect unexpected events and cause interruptions and jumps, for instance into box 8 or box 15.
- d) The chart does not indicate ways in which a path may have to be re-traced. For instance, while executing some plan one may reach box 5, then find that a required object is missing, which leads, via boxes 14 and 11, back to the top of the chart. The sub-process of box 11, namely recording the reason why the object is wanted and what is to be done with it when found (compare what was said above about the process-purpose index), ensures that when the object is later found or constructed there is a jump back to where the system was in the original process (box 5), so that it can continue, unless an embedded entry in box 9, has led to a revision of plans (and the process-purpose index) in the meantime. Further, if the object cannot be found, it may be necessary to go back to an even earlier phase, and perhaps choose another plan for which the missing object is not necessary. Thus, the possible jumps back to an earlier phase are not represented explicitly on the flow chart. [note 3]

6.11. Executive and deliberative sub-processes

It is perhaps worth noting the difference between the clockwise loops (on the left) and the anticlockwise loops (on the right). The former may be called *executive* loops, since they represent the kinds of processes which can occur when there is a plan of action and everything is going more or less according to plan. The anticlockwise loops may be called *deliberative* loops, since they represent the kinds of things that can happen when new planning is required, so that a question has to be answered, or an unexpected new obstacle or resource has turned up: the kinds of things which may require further intelligent deliberation and decision-making, using the agent's full resources.

The plans, or procedures, which generate uninterrupted executive processes may themselves have been stored or constructed only after previous processes of deliberation, involving many circuits round the anticlockwise loops. Even when things do not go wrong, there is always the *possibility* of dealing with difficulties and surprises, represented by the arrows going from left to right.

A system in which everything *always* worked exactly as described above would be much more efficient and rational than a human being. Nevertheless we know that human beings are often *capable* of doing the kinds of things the system can do, such as noticing unexpected obstacles and changing plans. The system does not therefore explain what people actually do; rather it generates, and thereby explains, a framework of possibilities which, for various reasons, may often not be actualised even though they would be appropriate, as in failure to recall a well-known fact or name. For reasons already mentioned, even a computing system of this kind must be fallible when it is very large.

6.12. Psychopathology

Notice that this outline of the structure of an intelligent mechanism gives enormous scope for analysis of various kinds of pathological conditions in which things go wrong. Indexes and catalogues may be destroyed or corrupted. Plans, procedures, and factual records may be destroyed or corrupted.

A spell in a peculiar environment may cause procedures and beliefs to be constructed which interfere with efficient functioning in other environments, and may be hard to erase or modify. The mechanisms which manage the purpose-process index may have faults. Monitors may fail to work normally, or else their 'something-found' messages may not reach appropriate destinations. A certain class of records may be intact, but the procedures for interpreting the symbols used may be faulty. Procedures for relating new information to the index of current processes and their purposes may be faulty. Good plans may be constructed, but mechanisms for executing them may be faulty. Alternatively, execution of available plans may proceed faultlessly, but the processes of constructing new plans may fail for one reason or another.

Various sorts of learning catered for in the above scheme may fail to occur. These are very general kinds of pathology. Other more specific kinds would require a quite different analysis.

Clearly, the task of interpreting and diagnosing pathological behaviour in such a complex system must be extremely difficult. It cannot be done without a good theory of the normal structure and functions of the system. This is why I have little faith in current methods of psychotherapy.

6.13. Conclusion: what is a mind?

This ends my sketch of the main features of a mechanism able to account for some of the main features of human thought and action, that is, able to answer a large number of questions of the form 'How is X possible?' In order to prove that such a mechanism is possible, it is necessary to design one in much more detail, filling in the form with much more content. The attempt to do this will probably

show up important kinds of hidden circularity, incompleteness or inconsistency in my description, leading to a revision of the specifications.

In order to demonstrate that this sort of mechanism provides an adequate explanation of the possibilities available to a human being, it is necessary either to analyse the specifications of the mechanisms and of the possibilities to be explained, and then prove mathematically that the mechanism does generate the required range of possibilities and nothing which it should not generate, *or else* to construct the mechanism and run it experimentally in a wide variety of circumstances to ensure that it produces an adequate variety of behaviour, with the required fine structure.

The former is likely to be well beyond the possibilities of mathematical analysis available in the foreseeable future, even though the mathematical analysis of programs and proof of their correctness is a developing discipline. In particular, it assumes that we can produce complete specifications of the possibilities to be explained, whereas one of the lessons of artificial intelligence is that attempting to design a working system often leads you to revise and extend your specifications. The experimental method may require the development of computers which have much faster processors and larger memories than at present.

Whichever approach is taken, it is necessary to have a good initial specification of the range of human abilities to be explained, and this is best achieved by combining philosophical techniques of conceptual analysis with the methods of social science and psychology.

Since each of the abilities makes use of many others, like a family of mutually recursive computer programs, there is no logical order in which they should be described: no ability is basic to the others. Further, none of them can be described completely without describing many others. This makes the task of constructing such descriptions, difficult, confusing and very frustrating.

The abilities which the above system is required to explain include:

- a) The ability to perceive and have perceptual experiences
- b) The ability to learn: skills, particular facts, general facts
- c) The ability to think about things including things near and remote, things previously met and new possibilities
- d) The ability to deliberate, decide, plan and act
- e) The ability to relate a purpose to available resources
- f) The ability to notice unsought-for facts
- g) The ability to reason, that is, the ability to use available knowledge to extend one's knowledge the ability to construct or manipulate symbols and representations, both verbal and non verbal, for such purposes as storing or communicating information, reasoning, deliberating, guiding actions, and so on

To specify these abilities in detail is to give at least part of an answer to the question: what is a mind? or what is a human mind? The partial answer is of the form: *a mind is something which can do such and such sorts of things*. To explain these abilities, that is, to explain how a single integrated system can do all these things, is to explain how it is possible for minds to exist. This does not merely make a contribution to the scientific study of man. It also brings many old philosophical discussions about the nature of mind and its relation to the human body several steps forward. (But it need not include

anything that Aristotle would have disagreed with.) In the process it is certain that many detailed problems in different branches of philosophy will be solved, rejected as confused, or brought nearer solution. The remaining chapters of this book address a few of the more detailed problems.

Endnotes

(1) An early version of this chapter appeared as *Memo 59* of the Department of Computational Logic, Edinburgh University, in 1972. A slightly revised version appeared in the *A.I.S.B. Newsletter*, February 1973. A much earlier version, called 'Chapter C' was circulated privately.

[[Note Added 1 May 2004

On re-reading this chapter I have become aware how much of my work over the last few decades has simply been elaboration and in some cases correction of the ideas in this chapter. Even the SimAgent toolkit, developed over the last 10 years to support work on architectures for agents with human-like capabilities has many features whose inclusion can be traced back to some of the requirements described in this chapter. The toolkit is summarised in:

<http://www.cs.bham.ac.uk/research/poplog/packages/simagent.html>
<http://www.cs.bham.ac.uk/research/cogaff/talks/#simagent>

Papers and slide presentations on architectures are here:

<http://www.cs.bham.ac.uk/research/cogaff/>
<http://www.cs.bham.ac.uk/research/cogaff/talks/>]]

(2) In the A.I. literature they are sometimes called *demons*.

(3) Flow-charts constitute a programming language. My remarks indicate that the language is too limited in expressive power. I never use them in my own programming, and do not teach students to use them, since careful layout in a language like LISP or POP2 (augmented with good iteration constructs) can achieve the same clarity without the same limitations.

[[Note added in 2001:

Two themes which are implicit in this chapter turned out to be important in later work, namely the role of real-time constraints in a fast-moving world, and the potential for a mechanism of the sort described here to get into an emotional state (See: A.Sloman and M.Croucher 'Why Robots Will have Emotions' in IJCAI 1981, available, along with other relevant papers at the [cogaff web site](#).)

The two themes are closely connected. The real-time constraints generate a need for various kinds of interrupt mechanisms, alluded to in this chapter. The potential for interrupts, which can disturb current activity is intimately connected with emotional states (in at least one of the many senses of 'emotional': some authors use the word so loosely as to cover all affective states including motives and attitudes.)]]

[Book contents page](#)

[Next: Chapter seven](#)

Last updated: 4 Jun 2007