Invited paper for "Information and understanding" session
at British Academy and Royal Society conference April 13th-14th 1994 on:
Artificial Intelligence and The Mind: New Breakthroughs Or Dead Ends?
To appear in *Philosophical Transactions of the Royal Society*

Semantics in an intelligent control system

Aaron Sloman
School of Computer Science & Cognitive Science Research Centre
The University of Birmingham
Email: A.Sloman@cs.bham.ac.uk

KEYWORDS

1. Semantic information
2. Artificial Intelligence
3. Cognitive Science
4. Mental Architecture
5. Control systems
6. Virtual machines

SUMMARY

Much research on intelligent systems has concentrated on low level mechanisms or limited subsystems. We need to understand how to assemble the components in an *architecture* for a complete agent with its own mind, driven by its own desires. A mind is a self-modifying control system, with a hierarchy of levels of control, and a different hierarchy of levels of implementation. AI needs to explore alternative control architectures and their implications for human, animal, and artificial minds. Only when we have a good theory of actual and possible architectures can we solve old problems about the concept of mind and causal roles of desires, beliefs, intentions, etc. The global information level "virtual machine" architecture is more relevant to this than detailed mechanisms. E.g. differences between connectionist and symbolic implementations may be of minor importance. An architecture provides a framework for systematically generating concepts of possible states and processes. Lacking this, philosophers cannot provide good analyses of concepts, psychologists and biologists cannot specify what they are trying to explain or explain it, and psychotherapists and educationalists are left groping with ill-understood problems. The paper outlines some requirements for such architectures showing the importance of an idea shared between engineers and philosophers: the concept of "semantic information".

1. INFORMATION, CONTROL AND AI

The questions posed were: *How does the engineer's concept of "information" differ from the philosopher's? In particular, can meaningful information exist without an understander?* I'll answer the former directly, the latter indirectly.

My claim, as a philosopher-engineer, is that a mind, or understander, is an *information-based control system* and all information within a mind is an aspect of one or more *control substates* in such a control system. Mental states and processes (such as beliefs, desires, fears) are *implemented* on the basis of information-bearing control substates and processes involving them, as are sophisticated computing systems. These control substates may be implemented using a hierarchy of levels of "virtual machines".[1]

This standpoint, links philosophy and engineering, solves the mind-body problem and many related philosophical problems (e.g. we can understand why *qualia* are needed in some self-monitoring control systems)[2], and, like any good theory, it opens up fascinating new research problems, e.g. problems about the architecture required to produce human-like control substates (e.g. desires and beliefs). Thus design problems replace many old philosophical problems.

This paper introduces some of the key concepts, sketches a subset of the links between philosophy and engineering relating to the "information level" description of behaving systems, and relates this to the aims of AI.

2. WHAT IS ARTIFICIAL INTELLIGENCE?

AI is misnamed, for its purview includes human and animal intelligence. I see it as *the general study of sophisticated self modifying information-driven control systems, both natural (biological) and artificial, both actual and possible (including what might have evolved or might be made).* This includes exploring architectures for information-driven systems, and a variety of implementation mechanisms, including connectionist and symbol manipulating mechanisms. AI can help us both to understand the world and to improve it.

Narrow, but all too common, views of AI, e.g. as a branch of engineering, or as a study of rule-based systems, do not do justice to the content of AI journals, AI conferences, and AI research laboratories. The full scope of AI is close to "cybernetics" as defined by Norbert Wiener (1948), whose pioneering book showed that he understood the centrality of information and the link between his new discipline and philosophy.

There are many traps for the unwary. One is to assume that we know what we mean by rich and complex, but inherently vague and ambiguous notions like "mind", "consciousness", "information" or "understander". Design-based theories will provide a basis for refining such notions.

A more subtle trap is to assume that our ordinary words mark *dichotomies*: namely sharp divisions between instances and non-instances, e.g. between understanders and non-understanders. Notions like "understanding" designate a very complex cluster of human capabilities. Different subsets may be found in different animals and machines, depending on their architectures and mechanisms, [3] and there is no "right" subset. Arguing about where to draw the line is pointless, but not because there is a *continuum* of cases: many design changes

[1] As explained in (Sloman 1978) and (Dennett 1991).

[2] (Sloman 1989, 1993b)

[3] (Sloman, 1985, 1986a, 1986b)

are *dis*continuous, e.g. adding an extra instruction to an algorithm. We need to explore the implications of the many subtle discontinuities in design space. We can then define new theory-based terminology in terms of different clusters of design features and capabilities.

This study of actual and possible designs may help us build new useful machines, but, more importantly, it also deepens our understanding of what we are, and provides a general framework for understanding the nature and variety of behaving organisms and how they evolved. That should lead to better ways of helping people through new forms of education, therapy or counselling. You can't repair something whose normal functioning you don't understand.

Wiener's vision of a mind as an information-based control system is not new. It can also be read into Freud, and even into Plato's view of a mind as analogous to a political system. What changes is the depth, precision and generality with which the idea is developed, and this depends on the set of conceptual tools available for thinking about the design and functionality of control systems.

Some versions of the idea risk circularity: by postulating capabilities in subsystems like those we are trying to explain for whole systems. A political model of mind is circular if the political system is composed of intelligent agents whose minds would therefore also be political systems composed of intelligent agents.

Early control theorists had non-circular, but very limited, conceptual tools associated with the mathematics then available for describing processes, for instance linear equations describing feedback loops. In the preface to his second (1961) edition, Wiener stresses the need for new conceptual and mathematical tools with broader powers. Computer science and AI were then in their pre-infancy. Though still infants in 1994, they have considerably enriched our notions of what sorts of mechanisms are possible, including mechanisms for manipulating complex structures with rapidly changing topologies, which cannot be accommodated by differential equations.

Using richer conceptual tools we can survey more architectures and mechanisms for self-modifying control systems and use their implications to generate a classification of types of behaving systems; and for each type of system a taxonomy of the types of states and processes that can occur. This is the "design-based" approach to the study of mind.[4]

Philosophers should be creative designers, helping engineers. The "design-stance" gives deeper insights than Dennett's (1978) "intentional stance" or Newell's (1982) "Knowledge level" analysis, since both assume rationality, and cannot account for the many departures from rationality due to different design goals, imperfect design, incomplete development, hardware malfunctions or software surprises. Much intelligent behaviour is not based directly on rational principles, but, for instance, on learnt heuristics activated by pattern matching under time pressure. It's in the design that we should seek rationality, not in the behaviour.

Wiener understood this sort of point: his chapter on psychopathology dealt mainly with abnormalities of perceptual and motor control due to neural damage or abnormality, but he had the insight that more subtle problems might arise out of incorrect information, including what we would now call "software bugs".

Our conceptual tools and our understanding of design requirements are still primitive. New theories about architectures and the functions they can support will generate more useful classification systems and taxonomies ("periodic tables" of mental states and processes), just as computer science extended our concepts of process and physics and chemistry extended our concepts of matter.

[4]It is described more fully elsewhere (Beaudoin and Sloman 1993, Sloman 1993a, 1994a, 1994b).

Exploration of design space, and the associated requirements space, or niche space[5] helps to solve or dissolve philosophical problems and to address questions like:
• Why and how do different capabilities evolve (such as the obscure collections of capabilities currently vaguely labelled as "consciousness", or "emotional states")?
• Which sorts of capabilities might have evolved in different environments or may evolve in the future?
• Theoretical questions about which sorts of capabilities various kinds of machines will have.
• Practical questions about how to build machines to perform particular tasks.
• Questions about departures from "adult normality" in human beings, for instance in children, in people with congenital brain defects or brain damage due to accident or disease, and, above all, in people whose motivational, or social or conceptual development has gone awry in some way that isn't due to physical or genetic factors.
• Practical questions about how to help in some of these cases.

Concepts of information are central to this exploration. There are different concepts of information, some shared between common sense, philosophy, science and engineering. I shall distinguish syntactic and semantic versions and show how the latter are common to notions of mind and to software engineering, and will try to bring out further links between design issues and philosophical questions, such as questions about the reality of information-processing levels and whether they have "real" causal powers.

3. "SYNTACTIC INFORMATION"

A notation, language, or representing structure has different aspects, sometimes referred to as syntax, semantics, pragmatics and inference. I have shown elsewhere[6] how these notions can be applied to different kinds of control substates, some not usually thought of as having a syntax or semantics, e.g. the "representation" of ambient temperature in a thermostat.

Syntax is concerned with the structure and forms of variation of a class of objects (sentences, pictures, signals or control states). *Semantics* involves reference to other things, in the environment, within the system, in the future, in the past, and possibly also things that have never existed and never will (like my olympic gold medal). I take *pragmatics* to cover the roles or functions of representing structures within an integrated system. *Inference* is concerned with the transformation of structures in a manner that usefully preserves or changes semantic properties or pragmatic roles.

We can discern two uses of the word "information" by engineers, one primarily syntactic and one semantic.

When dealing with issues of signal transmission, signal compression and detection and correction of transmission errors, engineers often use syntactic concepts of information, concerned mainly with the patterns in structures independently of whether those structures refer to anything outside themselves, as sentences and pictures typically do. One syntactic measure of "information" in a string is the length of the shortest program for a general purpose Turing machine which given any N returns the Nth symbol in the string. This defines "algorithmic complexity", an inverse measure of the amount of pattern or regularity in the string: the more regularity the less information. Logicians, linguists and computer scientists use concepts of syntax that are not concerned with *measures* but with *descriptions* of structures and their transformations.

[5] Sloman 1994a
[6] (Sloman 1994b)

4. "SEMANTIC INFORMATION"

The semantic concept of information, used informally by engineers, is closer to the familiar, though extremely ill-defined, notion of the "meaning", or "information" that may be conveyed by a message, or stored in a book or videotape. It is applicable to internal states as well as external communications. This includes ordinary concepts of mental states and processes such as "understanding", "believing", "desiring", "deliberating", "perceiving", "regretting", and many more. Philosophers call these states and processes "intentional" because they refer to something outside themselves, including, in some cases, nonexistent entities, e.g. hallucinating daggers, intending to perform actions which turn out to be impossible, or praying. Doing, preventing and trying are also semantic states, for they involve reference to future results.

All semantic information-processing depends ultimately on (internal) syntactic capabilities, for semantic information-processing depends on syntax-processing (i.e. structure-manipulating) engines, whether in computers, neural nets or other mechanisms. Agents need representational forms whose syntactic manipulation is semantically useful, unlike "languages" invented merely to illustrate the theory of syntax or parsing. A notation some of whose syntactically well formed formulae are semantically uninterpretable (like "Happiness eats democratic numbers intelligibly") is wasteful, though some meaning gaps may be important seeds of semantic development. Internal syntax is useful for processing information if it provides syntactic manipulations that map onto useful semantic relationships (e.g. syntactic derivability preserves truth). Much of the development of science and mathematics has been the creation of new formalisms that usefully compile semantic relationships into syntactic ones.

Computer programs have several kinds of semantics. One sort (called the "information level" below) refers to the application domain, usually outside the computer, e.g. information about employees. Another sort refers to the abstract datastructures manipulated at run time in the machine, e.g. numbers, strings, lists. Another refers to the low level machine instructions and memory locations manipulated when a program runs. Some languages also include compile-time semantics e.g. for macros, compiler directives or type definitions, which control compilation rather than subsequent running. Semantic information can vary both in content and pragmatic role. For instance it may be about what is the case, about what to do, about how to do things, or it may express a question or test. Which pragmatic roles are possible depends on the architecture.

Objects with semantic states include:

(a) humans and other animals,

(b) passive artefacts like books,

though the latter are often said to have only "derivative" intentionality, because their semantic states depend on the former. I shall argue below that

(c) active artefacts

such as factory control systems, office automation systems, are often more like (a) than like (b). Questions about the semantic states of biological control systems, information in DNA, chemical or neural messages, are also important, but will be ignored.

5. SEMANTICS IN CONTROL SYSTEMS

Engineers designing plant control systems, office support systems and the like, often start at the global information level, analysing semantic requirements for the whole system. For example a system may need information about the environment, rules and procedures to be followed, legal constraints, company objectives and which risks to avoid. Meta-information (information about information) is also needed, for example: where to get certain information, what to do when it arrives, how to cope with contradictory reports, and so on. Internal monitoring might yield meta-information about how quickly information is dealt with, which kinds turn out to be unreliable, and so on.

Designing information systems also raises *implementation* issues at different levels, such as:

- how information will be stored in the system,
- how to access it,
- selecting forms of representation,
- selecting syntactic transformations,
- selecting programming language(s) and operating systems,
- selecting computers, interfaces, network links, etc.
- functional decomposition of the system.

The semantic, or information level, specification, e.g. that the system must include information about employees and their roles and use it to perform certain tasks, says little about such implementational details. The specification can be given in an *implementation independent* way: including requirements for and the behaviour of a certain kind of information-processing *virtual* machine, leaving the computational or electronic details concerning "lower level" virtual or physical machines for later.

Moreover, implementation details may be revised as technology advances. Processors used, the memory technology, and even the programming languages and some of the low level algorithms may all change without implying any change in what information is processed, as far as the users and designers are concerned: i.e. the global information level description is not affected.

However, information level descriptions may imply a certain sort of *architecture*: a top level functional decomposition, defining which sorts of major subsystems coexist, and which information they handle. For instance, being undecided about whether to go to A or to B, presupposes mechanisms for manipulating goals, for evaluating and selecting between alternatives, and for acting on a selection.

For subsystems we can also define an information level: what they "know" about the rest of the system, i.e. *their* environments and their tasks. Typically, users will not be concerned with that, though designers and maintainers will.

6. IMPLEMENTING INFORMATION STATES

Our understanding of how one machine "implements" another is still mostly intuitive, for we lack good *general* theories and terminology. Nevertheless, there is no mystery: we can make it happen.

A working system always has a physical level of description. Phenomena at other levels may be "emergent" in that concepts needed to describe them cannot be defined in terms of the lower levels, and the laws of behaviour of higher level virtual machines are different from

and not derivable from the laws governing the lower implementation levels. Information level concepts like "refers to" or "attempts to", or, for that matter, "customer", cannot be defined in terms of mass, velocity, voltage, etc., and neither can the information level principles of behaviour be derived from physical laws: e.g. they may depend on business practices, legal constraints and the social environment, which change while physics doesn't. Mental states may depend on a culture, e.g. physics doesn't determine musical style or acceptable grammar. Thus information level specifications enrich our ontology with new concepts and new laws, relative to physics.

Several levels of abstract "virtual machines" can coexist in a single system, each with their own information level characteristics. In a word-processor one abstract machine performs operations on chapters, sections, paragraphs, words, footnotes, and so on. It may be "implemented" in terms of another that manipulates arrays, strings, lists, and other datastructures, corresponding to "high level" programming languages. Below that level is a virtual machine defined in terms of operations on bit-patterns, describable in the computer's "machine code". Still lower levels consist of abstract digital machines, described in circuit diagrams. Below that are physical states of components, describable in the language of physics.

There is no well defined "bottom" level. The lowest level actually considered depends on the task. For many software engineers lower layers are irrelevant (except when things go wrong), though for compiler writers and hardware designers they are crucial. Sometimes close-coupling between high and low levels is useful, e.g. microcode instructions invoking high level procedures, quantum-based randomisers, and perhaps chemical soups for global control in future computers.

Normally concepts at each level are not *definable* in terms of the concepts at a lower level. Word-processor concepts, like "page" or "sentence" are not definable in terms of concepts of physics, nor in terms of arrays, lists or strings. Page-formatting rules are not deducible from physics, nor from equations defining data-types. Each level defines an emergent ontology, with its own laws.

7. IMPLEMENTATION AND SUPERVENIENCE

Many have likened the relationship between an information level virtual machine and lower levels in a software system to the relationship between mental processes and the physical brain mechanisms implementing them. Mental descriptions, like "believes" and "desires", are used in ignorance of implementation details, just as information level descriptions of software systems are usable by people who know nothing about the programming languages, datastructures and algorithms, or underlying electronic mechanisms.

Some information states involve relationships to external objects and therefore cannot be implemented solely in terms of *internal* states. A computing system cannot store information about Joe Smith *solely* in virtue of internal states: how the system is related to that external individual also matters. Similarly my beliefs about Joe cannot be implemented entirely in my brain states. Such information level states have "intrinsically relational content", and the environment provides part of their implementation. (Some philosophers label this "wide content" or "broad content".) Thus two physically identical systems in different locations will not necessarily contain absolutely identical information.

Particular lower level states are not necessary for the high level states, if alternative implementations are possible. Neither are they sufficient, if successful implementation

depends crucially on the current environment, like reference to Joe.

In philosophical terminology, mental phenomena are "supervenient" on physiological or physical phenomena. Similarly, computer-based information states are supervenient on physical phenomena (both internal and external). This philosophers' relation is simply the converse of the engineer's relation of implementation. (Both need to accommodate intrinsically relational content.) Philosophers grappling with "emergence", or "supervenience" (e.g. Robinson 1990, Horgan 1993) might be helped by software engineering examples which are already well understood.

Animal or machine control systems, including human minds, have an underlying physical implementation, whether based on transistors, neurons, or rotating wheels. But this does not imply that different items of information must be mapped onto physically *separable* components (physical symbols). High level virtual machine details (including semantic content) need not correlate with or map onto physical structures. Distinct items of information can be superimposed on electromagnetic waves and later separated out using filters. Neural nets allow information items to be superimposed and distributed over a collection of synaptic weights, or over activation levels of a set of neurons. Two numbers can encode the row, column and both diagonals a chess piece is on – four items of information superimposed on two. Huge 'sparse' arrays in a virtual machine include far more components than the physical structures that implement them. A set of axioms can encode, in a distributed and superimposed form, an indefinite variety of different items of logically derivable information, and which information is extractable can vary with context or time-pressure. Finally, relationships with the environment may help to determine content.

In such cases, examination of internal physical structures will not reveal information systems they implement: for much of it is *implicit* in how substructures are used by other components.

Some implementation details have little impact on overall functionality: they make a marginal difference (e.g. to speed, or to reliability under high temperatures that rarely occur) or a big difference only in rare situations (e.g. getting most common questions right). Whether particular subsets of an architecture use neural nets or some other mechanism may be marginal in relation to the functioning of the whole system. It is global design that matters most: which subfunctions are provided, how they are linked, how they are used and how they change. *Architecture dominates mechanism.*

8. LIMITS OF TRANSLATABILITY

An architecture and the mechanisms involved do not determine information content: as shown by the possibility of the same computing system containing different information at different times. But system characteristics may *limit* the possible semantic states, for at least two reasons:

• Variability in the mechanism or notation may not match the variability in the semantic content, just as not all the information in human mental states is fully expressible in external or internal sentences, e.g. one's knowledge of someone's face, how strawberries taste, or the appearance of swirling rapids in a fast flowing river. (J.L. Austin once wrote: "Fact is richer than diction").

• The architecture may not support the required functionality. E.g. phrases like "believes that...", "perceives that...", may not fit the pragmatics of control states of organisms with non-human architectures.

This implies that there are limits to semantic translatability. If an animal has a perceptual, or motivational, or cognitive state, humans may be incapable of experiencing or imagining those states, because those states exist in an architecture too different from ours. Thus what it is like to be a bat, or a new born baby, or a robot of the future may not be expressible in information structures of an adult human brain. Or vice versa. This supports Wittgenstein's remark that if a lion could talk we would not understand him. (It's hard enough with some of our own kind.)

This sort of problem may limit replication of certain human states in computer-based robots. Likewise, implementation-level differences between brains (e.g. adults and infants) might rule out identical mental states among humans. Nevertheless if we don't demand *exact* correspondence, it may be possible to replicate many aspects of human information states in meatless machines. Which, and how, remains a research problem. (The so-called Strong AI thesis, claiming that it is possible, breaks down into at least eight different theses, as I've shown in (Sloman 1992).)

9. INFORMATION AND CONTROL

What is special about information-processing systems? Do trees and clouds store and use semantic information? Are they understanders, and if not why not? As remarked in section 2, "understanding" refers ambiguously to a *cluster* of capabilities with no unique defining subset. Nevertheless we can make a rough distinction between mechanisms controlled only by physical laws and mechanisms controlled by virtual machines involving information. The latter involves explicit prior representation of possible actions prior to their execution.

Examples are systems containing two or more control substates capable of producing different behaviour (internal or external) between which selections are made, for example by examining some other substate of the system. Simple examples include computing systems that support instructions like:

```
if <condition> then <action1> else <action2>
```

In more sophisticated cases the `<actions>` are parametrised, with details filled in as a side-effect of testing a `<condition>`. (That usually requires lower level conditionals to have explicit alternatives.) What is then explicit initially is a *schematic* action specification, with building blocks for *creating* the final specification in advance of performing the action. By contrast, movement of a cloud or a tree depends on external and internal conditions but is not controlled by selections from or construction of explicit prior representations of options.

A slight generalisation accommodates connectionist systems: they exhibit *parallel* and *cumulative* conditional influences, and some depart from simple binary logic. The pattern of firing of a set of neurons typically depends on

- the previous pattern in some other (possibly overlapping) set,
- the current weights on the connections between the two sets,
- a decision function for combining weighted influences.

This is also conditional control, but all conditions are *tested in parallel* and the outcomes are *sent in parallel* to representations of possible actions, which are triggered or deactivated on the basis of the *total accumulated* support or inhibition they receive. Such networks may be clocked or asynchronous, based on discrete or continuous variation (e.g. of weights or activation levels), and may use different numbers of coexisting layers and connection topologies (e.g. cyclic or acyclic). These are all variants of the general notion of information-based control, as are probabilistic or fuzzy logic systems. However, we should

not expect a sharp and total division between systems that are controlled by information and those that are not. This is another "cluster concept".

Among the useful features of information-based control is *context sensitivity* i.e. producing different behaviours on different occasions in response to *the same* specific (internal or external) stimulus, because links between cause and effect are indirect insofar as they depend on conditionals.

How a condition works may change conditionally under hierarchic control. How A influences B can depend on information structure C. How it depends on C can depend on D (e.g. if D changes the tests in C, re-arranges the order of testing or alters connection weights). The exact role played by substates of C, and how the information is used to change B, can vary enormously from moment to moment, under the influence of other information states. How D influences the testing can depend on E, and E's influence on D may be modified by F, and so on. For instance the length of time a program P runs can depend on the scheduler S, and which files P may access depends on the file manager, M, where both S and M can be modified by an administrative tool, or may even be modified by the operating system itself using performance statistics.

In contrast with animal brains, artificial information-based control mechanisms are well understood. In simple cases programs access information stored in memory, or in input registers, and what they find determines selection of actions. A more abstract virtual machine in a vision system may at one instant interpret an intensity discontinuity in an image array as a crack in a surface, then moments later as an edge of a surface or as a stretched string. The resulting actions will be different.

In such cases, how A affects B is not a physical law. Compared with control by physical (e.g. mechanical or electrical) influences, information-based control:

• is more flexible (e.g. can be more sensitive to changing context),
• admits more rapid change of control relationships,
• allows more superimposed layers of dispositions concerned with different functions with different time scales,
• permits effects to be saved up for future use (so that feedback loops need not have fixed time constants),
• supports teleology of a kind once thought mysterious: namely control by representations of future objects that don't yet exist, including pipe-dreams that never will.

These features depend on different internal states having different control functions and embodying different sorts of information. This functional differentiation of control states is what I call (high level) architecture. In general it will not map onto physical architecture in any simple way. That is one reason why the evolution of biological control capabilities is so hard to study: virtual machines leave no fossil records.

10. HUMAN-LIKE INFORMATION STATES

What sorts of virtual machine architectures are required for human-like, chimp-like, or squirrel-like agents? Autonomous agents apparently need a coarse-grained high-level functional division into components with different but interlocking functions, such as perception, motor control, various short term and long term information stores, inference systems, and planning systems. Humans seem to use a complex control hierarchy of *motivational* states, dealing with different levels of abstraction (e.g. personality, attitudes,

desires), different time scales and different forms of decision-making.[7] There are also problems about resource limits, especially in high-level "management" processes, where interrupt-driven attention switching is often needed to cope with new problems or motives, but which sometimes need to be protected from such diversions, perhaps using context sensitive filtering. These architectural issues are closely bound up with notions of self-control and partial loss of control, e.g. in some emotional states.[8]

Some human semantic states (e.g. beliefs) support notions of "truth" and "falsity". These involve both a semantic relation between representation and reality and pragmatic roles in selecting actions. For computing systems with two-branch conditionals, truth of the `<condition>` is simply whatever value makes the system perform `<action1>` rather than `<action2>`. If the process of evaluating a condition is liable to error, and various checks and alternative methods are available, the machine can treat truth and falsity as involving something accessible via different checking routes. Information may then be more or less complete, more or less detailed and more or less accurate. A human-like concept of truth would require a further level of self-knowledge that implied that *any* current checking method could turn out to be inadequate and require replacement.

Since conditions may be false more often than true, meaning does not depend on correlations with reality.

Specifying high level architectures to support human-like semantic states is no trivial matter. Our approach is cyclic: analysing a variety of human scenarios (including motivational and emotional processes), attempting to extend the current architecture to account for them, then discovering that it cannot cope with further scenarios, so that the architecture has to be further enriched or replaced.[9] This process, which has much in common with philosophical analysis[10] can be combined with "bottom up" physiological and physico-chemical investigations, evolutionary studies, and the like.

11. THE TURING TEST

The design stance is important because very different internal processes can produce the same input-output relationships. This makes it impossible to infer internal information processes simply from externally observable behaviour. One system may explicitly consider alternatives, evaluate them, select one and act on it, and possibly learn from the result, while another behaves the same way (externally) because *its designer* thought about all the questions that might arise, considered the alternatives, and pre-programmed the answers. This is often referred to as simulating intelligence by means of a "huge lookup table" (HLT). Software engineers often convert a program from one of these forms to another, e.g. trading compactness and generality for speed.

Since all we have to go on in judging mental states of others is behaviour, some people think that concepts of mental states and processes are definable simply in terms of behavioural capabilities and dispositions of the *whole* system, independently of how it works internally, e.g. what its architecture is. Similarly the 'intentional stance' ignores internal processing. If intelligence depends on *how* behaviour is produced this must be wrong.[11]

[7] (Sloman 1993b)

[8] (Beaudoin & Sloman 1993)

[9] (See Sloman 1985, 1986b, 1993a,b, Beaudoin 1993)

[10] (Sloman 1978, ch. 4)

[11] Turing himself was too intelligent to propose passing his test as a criterion for intelligence or understanding:

Philosophers should go beyond global capabilities, and behavioural criteria, and adopt the design-stance if they wish for a deep understanding of ordinary mental concepts. If I am wrong and ordinary concepts are not design-based, then so much the worse for them: design-based concepts are important for full understanding of how our minds work and how they may fail to work, or work inappropriately. New theories about how minds work will cause our mental concepts to evolve, just as modified concepts of kinds of stuff grew out of theories about the architecture of matter.

12. INFORMATION STATES AS CAUSES

Why do I call abstract systems "machines"? Because they have states and processes and causal interactions. The common objection that "real" causation is possible only between physical events[12] ignores the following:

• We often talk about causal connections between non-physical events: noticing someone's expression can cause one to suspect his motive, and changes in the tax law can make one much poorer.

• The concept of causation is still not well understood, and may be systematically ambiguous, as described in (Taylor 1992).

• The objectors presuppose a well-defined bottom physical layer of reality, whereas progress in physics leads to ever more subtle and mysterious notions that seem to have little to do with our ordinary concepts of causality.

• Software designers commonly talk about causal connections between events in virtual machines. E.g. "These instructions caused the words to be sorted into reverse alphabetical order." "Information about receipt of an information packet causes another to be sent." "A syntactic error in a program caused compilation to terminate." These all describe causal relationships between events or processes in virtual machines. They allude, in a subtle way, to counterfactual conditionals about what would or would not have occurred had something been different. Of course there are underlying physical causal processes. But to claim that *only* the physical causes are real causes would imply that software engineers cannot do their job without becoming electronic engineers, or perhaps quantum theorists. This is clearly absurd: they can and do set up desired causal connections, diagnose unwanted causal connections, and explain the behaviour of complex systems, all at the information level.

I conclude that there is no reason for philosophers, engineers, or ordinary folk, to "eliminate" talk of information states and processes as having causal powers. Nevertheless, better theories of the virtual machines involved, will revise and extend concepts of human mental phenomena.

13. NON-DERIVATIVE SEMANTICS

Another objection, using the distinction introduced in section 4, states that intentionality in computers is "derivative". When we say that a person is worried about poverty, or that a book is about poverty, both involve semantics, but the semantic properties of the book are "derivative" because they depend on something else to understand the contents, whereas a person does not require others to understand his state of mind.

---

he merely offered it as a technological challenge he thought could be met without using an HLT.

[12] Robinson (1990) surveys the arguments.

Some philosophers claim that active artefacts, like computing systems, are no different from books, since both contain information, but not information that they can understand: computers are just automated filing cabinets, and their states have only "derivative" semantics.

It is true that many computer databases no more understand their contents than a book does – they manipulate, but do not use, the information. But this misses a subtle point.

Although the mathematical concept of "computation" is purely syntactic (Sloman 1992), working computers are not purely syntax manipulators. Their usefulness depends on their ability to interpret as well as manipulate symbols. A computer searching a list for words ending in 'ing' uses information about where the contents of the list are, about the words to be selected, and about what to do with them. At a lower level it uses machine instructions and information about memory locations, both of which it understands, albeit in a primitive fashion.[13] Self-tuning operating systems can assemble and use information about current and past processes that is known only to and used only by the computer. Robots and plant controllers can use information about the environment.

Mutual dependencies within an architecture bootstrap a collection of mere mechanisms into an understander: what sort of understander depends on the architecture.

Current machines are a long way from human semantic competence. Overcoming this (for practical or theoretical purposes) requires us to understand the variety of mental functions of which a human being is capable, and learn how to put them together, including motivational processes missing from most existing AI systems.

It is often assumed that non-derivative semantics (or "symbol grounding") requires the architecture to be embedded in and connected to a physical environment via sensors and motors. By definition, such embedding is required for intrinsically relational information states (e.g. beliefs about Joe), and the majority of human mental states may well be like that. But not all reference involves the environment. Many mental states, including beliefs, desires, hopes, plans, joy, despair, and more, could, in principle, occur in an appropriate architecture concerned *entirely* with the internal exploration of number theory. (People who have never experienced mathematical research may not understand!)

Some philosophers claim that unless a machine is the product of biological evolution its internal states and processes cannot be beliefs, desires, and so on. This is just linguistic imperialism. If such people restrict certain words to connote having a biological history, there's nothing to stop the rest of us using them for closely related concepts that apply to objects with similar capabilities and different histories: products of laboratories, not evolution. If machines work in a manner that is sufficiently like us then by using the ahistorical connotations we can save ourselves the trouble of inventing a whole new vocabulary for describing and communicating with them. As Young (1994) states, it's not history but the potential for future actions that matters.

14. CONCLUDING COMMENTS

The intuitive concept of semantic information or "aboutness" is part of common sense and also philosophy and engineering. It applies to information states and processes as opposed to physical and physiological states and processes. Information states enter into causal interactions and control functions in high level virtual machines, enabling sophisticated and flexible internal and external behaviour.

[13] This is elaborated in (Sloman 1985)

We currently understand a lot about causal connections in information level virtual machines in computing systems. This provides clues about how to think about causal connections in the high level virtual machines constituting human minds, and may eventually lead to an improved theory-based classification of mental states, just as understanding the architecture of matter gave us improved concepts of kinds of stuff. E.g. our muddled ideas about "consciousness" including the prejudice that "conscious states" are *intrinsically* different from others may prove merely to rest on a distinction between states that are and states that are not accessible by certain high level self-monitoring processes.[14] Some states and processes that are inaccessible to consciousness may be *exactly* like accessible ones, and the boundary can change through training.

Many unanswered questions remain. E.g. how and why did different information-processing capabilities evolve? This breaks down into several questions. How did different forms of syntax evolve? How did different functional roles (pragmatics) for control substates evolve? How did different semantic capabilities evolve? How did different inference capabilities evolve? And how did architectures evolve that integrate all these capabilities in a multi-functional whole? Whether computer-based systems could replicate high level functionality of natural brains is also open.

Answering these questions requires AI theorists to survey and classify the variety of information-bearing states to be found in human beings, other animals, and machines of various sorts. We need to study "dimensions of sophistication" in which architectures can differ, including: the number and variety of concurrent high level functions, the variety and complexity of forms of syntax and structure manipulation used in information stores and control states, the flexibility and depth of perceptual processing, the variety of sources of motivation, different kinds of internal self monitoring, different kinds of self control and internal management, varieties of self-modification and learning, including modifications of the architecture itself by addition of new capabilities or mechanisms and new links between old ones. These themes are developed in papers listed below.

This "exploration of design space"[15] has barely begun, even though some AI researchers prematurely commit themselves to particular representations, e.g. logical or connectionist, or to restrictive architectures.

Because the tasks are so difficult, AI may look like a "dead end" to those who do not understand the variety of important but difficult problems that remain. A subject with so much important unfinished business cannot be at a dead end. But we should not insist on some narrow set of explanatory tools and concepts. That would be as silly as trying to restrict physics to the concepts and mathematics that Newton understood. Infant disciplines must be allowed to grow.

Many philosophical views are based on emotional commitments and cannot be changed by evidence and rational discussion. In the long term, results of design-based investigations may convince some doubters, but never all. For some, theoretical analysis will suffice. Some will change only after developing personal relations with intelligent androids. Others may be convinced when the new theories help us solve difficult human problems, for example in education and therapy, which, at present are neither science nor engineering but often hit-and-miss craft activities.

[14](Sloman 1978 ch 10)

[15]and the niche space described in (Sloman 1994a).

ACKNOWLEDGEMENTS

I am grateful for comments on an earlier draft from: Alan Bundy, Edmund Robinson, Robert Kirk, Yves Kodratoff, Peter Lupton, Neil Rickert, Roger Young. I have learnt from the work of too many others to cite them all. This work is supported by a Joint Council Initiative grant, MRC code SPG9200393, and the Renaissance Trust.

REFERENCES

Beaudoin, L.P. and Sloman, A, 1993, A study of motive processing and attention, in A.Sloman, D.Hogg, G.Humphreys, D. Partridge, A. Ramsay (eds) *Prospects for Artificial Intelligence*, Amsterdam: IOS Press, 229-238,

Dennett D.C. (1978). *Brainstorms: Philosophical Essays on Mind and Psychology* Cambridge, MA: MIT Press.

Dennett D.C. (1991). *Consciousness Explained* Allen Lane, the Penguin Press.

Gibson, J.J. 1979, *The Ecological Approach to Visual Perception* Lawrence Earlbaum Associates, re-issued 1986, New Jersey: Hillsdale.

Horgan, T, 1993, From Supervenience to Superdupervenience: Meeting the demands of a material world, *Mind* Vol 102, No 408, 555-586

Newell, A, 1982, The knowledge level, *Artificial Intelligence* Vol 18 no 1, 87-127.

Robinson, W.S. 1990, States and Beliefs, *Mind* Vol XCIX No 393, 33-51.

Sloman, A, 1978 *The Computer Revolution in Philosophy: Philosophy Science and Models of Mind,* Hassocks: Harvester Press

Sloman, A, 1985, What enables a machine to understand? in *Proceedings 9th International Joint Conference on AI Los Angeles* 995-1001.

Sloman, A., 1986a, What sorts of machines can understand the symbols they use? in *Proceedings of the Aristotelian Society*, Suppl. Vol. LX, 61-80.

Sloman, A. 1986b, Reference without causal links in *Advances in Artificial Intelligence - II* eds J.B.H. du Boulay, D.Hogg, L.Steels, North Holland, 369-381.

Sloman, A., 1992 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, 355-396

Sloman, A 1993a, Prospects for AI as the general science of intelligence in A.Sloman, D.Hogg, G.Humphreys, D. Partridge, A. Ramsay (eds) *Prospects for Artificial Intelligence*, Amsterdam: IOS Press, 1-10.

Sloman, A. 1993b, The mind as a control system, in *Philosophy and the Cognitive Sciences,* (eds) C. Hookway and D. Peterson, Cambridge University Press, 69-110.

Sloman, A. 1994a, Explorations in Design Space, to appear in proceedings *11th European Conference on AI* Amsterdam 1994.

Sloman, A, 1994b, Representations as Control Substates *in preparation*

Taylor, C.N, 1992 *A Formal Logical Analysis of Causal Relations* DPhil Thesis, Sussex University. Available as Cognitive Science Research Paper No.257

Wiener, N. 1948, revised ed. 1961, *Cybernetics: or Control and Communication in the Animal and the Machine* 2nd ed. Cambridge, Mass.: The MIT Press (1st edition 1948)

Young, R.A. 1994, The Mentality of Robots, *Proc. Aristotelian Soc.*