A design study for an Attention Filter Penetration architecture

Brian Logan
School of Computer Science & IT,
University of Nottingham, Nottingham UK.
bsl@cs.nott.ac.uk

Abstract

This paper describes a design study for a variant of the 'Attention Filter Penetration' (AFP) three layer architecture (Sloman, 2000). In an AFP architecture, the activities of an agent are distributed across three concurrently executing layers: a *reactive* layer in which detection of internal or external conditions immediately generates new internal or external response, a *deliberative* layer responsible for 'what if' reasoning and planning capabilities, and a *meta-management* layer which provides self monitoring, self evaluation and self-redirection including control of attention. Our design is based on two main assumptions: that deliberation (and meta-deliberation or management) is the technique of last resort for an agent, and is only used when no reactive behaviour is clearly applicable in a given situation; and deliberation is just something that some reactive systems do. The paper attempts to identify those parts of the design which seem fairly uncontroversial, and to highlight those issues which have yet to be resolved.

This paper describes a design study for a variant of the 'Attention Filter Penetration' (AFP) three layer architecture (Sloman, 2000). The aim is to design an architecture for an agent which can control its efforts to achieve one or more goals in some domain, and for the system to be able to adapt its behaviour to changes in the environment and difficulties encountered in achieving its goals.

In an AFP architecture, the activities of the agent are distributed across three concurrently executing layers: a *reactive* layer in which detection of internal or external conditions immediately generates new internal or external response, a *deliberative* layer responsible for 'what if' reasoning and planning capabilities, and a *meta-management* layer which provides self monitoring, self evaluation and self-redirection including control of attention. An attention filter with a dynamically varying interrupt threshold protects the resource-limited deliberative and meta-management layers when dealing with tasks that are important, urgent and resource consuming.

We make two main assumptions: that deliberation (and meta-deliberation or management) is the technique of last resort for an agent, and is only used when no reactive behaviour is clearly applicable in a given situation; and deliberation (and management) is just something that some reactive systems do. In doing so, we are not attempting an explanatory reduction the deliberative or management layers to the reactive layer, rather the aim is to sketch an implementation of the virtual machines which operate at these layers in terms of the primitives available at the reactive layer. Some such reduction must be possible: some kinds of deliberative and management behaviour must 'just happen' otherwise we end up with infinite regress. However there is no reason in principle why they should reduce to mechanisms at the reactive layer, they could, for example, be implemented using distinct machinery 'at' their respective layers. The assumption that they are not is perhaps the central design decision of this paper.

Of particular interest therefore is the interaction between the reactive and deliberative layers: both the generation of new motives or goals by the reactive layer and when and how these are scheduled for processing at the deliberative layer. We focus first on the reactive layer of the architecture to clarify what it can and can't do, and to outline how it does what it does, before attempting to show how the deliberative and management layers can be implemented as particular kinds of reactive behaviour.

The paper attempts to identify those parts of the design which seem fairly uncontroversial, and to highlight those issues which have yet to be resolved. In several cases, a number of possible approaches to an unresolved issue are identified. Such speculations are not intended as exhaustive enumerations of the options, rather they attempt to indicate the current state of work on a topic and illustrate insofar as this is possible at this stage, some of the main issues that would have to be addressed by any solution.

1 The Attention Filter Penetration architecture

In this section, we briefly describe the Attention Filter Penetration three layer architecture which forms the basis of this study.

The Attention Filter Penetration architecture attempts to account for the existence of a variety of more or less sophisticated forms of information processing and control in human and other minds. The version discussed here is based on previous work by Sloman and others (Sloman and Croucher, 1981; Beaudoin, 1994; Sloman, 1994; Sloman and Poli, 1996; Sloman, 1997, 1998, 2000; Sloman and Logan, 1999) and postulates three concurrently active layers which evolved at different times and are found in different biological species. The three layers account for different sorts of processes. None of the three layers has total control: they are all concurrently active and can influence one another.

The first layer contains *reactive* mechanisms which automatically take action as soon as appropriate conditions are satisfied. The second *deliberative* layer provides 'what if' reasoning capabilities, required for planning, predicting and explaining. The *meta-management* layer provides the ability to monitor, evaluate, and partly control, internal processes and strategies.

Roughly, within the reactive layer, when conditions are satisfied actions are performed immediately: they may be external or internal actions. A reactive system may include both analog components, in which states vary continuously, and digital components, e.g., implementing condition-action rules, or various kinds of neural nets, often with a high degree of parallelism.

By contrast, the deliberative layer, instead of always acting immediately in response to conditions, can contemplate possible actions and sequences of possible actions (plans), compare them, evaluate them and select among them. The human deliberative systems can also consider hypothetical past or future situations not reachable by chains of actions from the current situation, and can reason about their implications. Deliberation requires a large amount of stored knowledge, including explicit knowledge about which actions are possible and relevant in various circumstances, and what the effects of various actions are in those circumstances. It also requires a re-usable short term memory for building structures representing possible action sequences in order to evaluate their consequences. The reuse of memory, the inability of the long term store to answer many questions in parallel, and the sequential nature of plan construction will typically make a deliberative system much slower than a reactive one.

The meta-management layer acts on some of the internal processes involved in the reactive or deliberative (or meta-management) system. This includes monitoring, evaluating and redirecting such internal processes, and possibly reflecting on them after the event in order to analyse what went wrong or how success was achieved. Like the deliberative layer, it will be resource-limited.

Both in a sophisticated reactive system and in a deliberative system with planning capabilities there is often a need for motives which represent a state or goal to be achieved or avoided. In simple organisms there may be a fixed set of drives which merely change their level of activation depending on the current state of the system. In more sophisticated systems not all motives are permanently present, so there is a need for *motive generators* to create new goals possibly by instantiating some general goal category (*eat something*) with a particular case (*eat that foal*). These generators are similar to the dispositional 'concerns' in Frijda's theory (Frijda, 1986). Beaudoin (Beaudoin, 1994) and Wright (Wright, 1997) discuss various types of generators or 'generactivators' and related implementation issues.

Since the different layers operate concurrently, it is possible for new information that requires attention to reach a deliberative or meta-management sub-system while it is busy on some task. Because of resource limits, the deliberative sub-system may be unable to evaluate the new information while continuing with the current task. However it would be unsafe to ignore all new information until the current task is complete. New information therefore needs to be able to interrupt deliberative processing.

Under stressful conditions, deliberative mechanisms with limited processing or temporary storage capacity can become overloaded by frequent interrupts. Beaudoin & Sloman (Beaudoin and Sloman, 1993) have argued that a variable-threshold attention filter can reduce this problem. Setting the threshold at a high level when the current task is urgent, important and intricate, can produce a global state of 'concentration' on that task. Conversely, malfunctioning of this mechanism may produce a type of attention disorder (Beaudoin, 1994).

2 The reactive layer

In this section we present some observations about the nature of reactive behaviours and sketch a design for the reactive layer of a simple agent.

We assume that the agent has some simple behaviours which are triggered by the presence or absence of certain features in the environment. Some of these behaviours are atomic, whereas others are composed of simpler behaviours and may have complex internal organisation such as conditionals, loops etc. which *implicitly* anticipate the future. In general behaviours have duration; some behaviours are effectively instantaneous, but for those that are not, it is often more natural to talk in terms of an action such as *move forward* rather than a sequence of equivalent actions, such as *move forward 1m*, each of which can be accomplished within some time bound. In many cases, multiple behaviours can run in parallel. Some behaviours which can't simply be executed in parallel, for example because they require the use of one's hands, can however be 'blended'; in other cases the activation of two behaviours for the same processor or which require the same 'resource' may give rise to a explicit goal to choose between them (see below).

Usually such behaviours are completely automatic; the presence of their triggering condition(s) in the environment will always elicit the behaviour. However, the execution of these behaviours is subject to various forms of control. Some 'reflex' behaviours can be temporarily suppressed or their execution modified as a result of feedback from the environment. In some cases, complete suppression of the behaviour requires 'prior notice', e.g. not blinking while undergoing an eye examination, though some modification is often possible even after the behaviour has been initiated, e.g. stifling a cry of alarm or surprise. Other forms of control appear to be global modifiers of all behaviours at the reactive layer, for example when late for an appointment we may perform all routine actions hurriedly, or aggressively when we are frustrated. Such control also extends to modifying the expression of a voluntarily executed behaviour or set of behaviours, as when we perform an action 'carefully', 'quietly', 'theatrically' etc. In this case, it is not clear if the control is accomplished by 'stepping through' the basic actions of the behaviour in such a way as to change their subsequent execution, or whether we actually synthesise a new 'careful' version of the behaviour 'on the fly' from more basic behaviours.

It is convenient to model reactive behaviours as sets of condition-action rules. The condition part is matched against the agent's perception of the current world state, and, if it fires, it triggers a single (ballistic) action which attempts to change the state of the world in some way. Such a behaviour is entirely stateless, in that it maintains no internal representation of the state of the world, whether the rule has been fired before etc. However, even such basic 'reflex' behaviours can be chained together to produce quite complex behaviour, for example, if we (or evolution) arrange things so that one action changes the world in such a way as to trigger another action, and such an architecture can support conditionals and simple loops.

One problem with this approach is that if any action fails to achieve its 'intended' result, then the whole sequence of actions may fail unless: (a) the action had no significant effect, allowing the rule to be re-invoked in the hope that this time it will succeed; or (b) the action has some other effect, recognisable as an 'error condition' and there is an 'error recovery rule' that can get things back on track for the 'intended state'.

Another problem is that the agent will always respond to the same situation in the same way, and will continue to respond in the same way if its efforts to change the world are unsuccessful. One way round this is to arrange for the rules to match against some internal representation of the environment, rather than percepts directly generated by the agent's sensors. This allows the agent (or some of the agent's behaviours) to respond only to *changes* in the environment, ignoring those features that are constant (without some representation of the previous state, we can't say what is novel in the current state). One way to build such a representation is to use simple percept driven condition-action rules to record simple 'beliefs' about the state of the world. Such rules perform a very primitive kind of belief fixation, in which the 'beliefs' are simple flags in the agent's internal state which correspond one to one (modulo sensory noise and other errors) with the world as the agent perceives it.[1] Other rules match against this internal representation, and, if they fire, trigger actions. From the point of view of expressive power, this adds nothing. We have simply taken a condition-action rule and split it in two, with a mediating internal representation. To detect and represent changes in the environment, we also need rules whose conditions match against the agent's internal state and whose actions modify the agent's internal state.[2] Given such internal behaviours, much more complex external behaviours are possible.

More complex reactive behaviours also require additional internal representations to hold intermediate state during their execution, for example unachieved implicit goals or the current state of a task (where 'state' is understood as a predefined stage in a task, rather than some general representation of the world). This includes negative feedback loops (where there is at least an implicit representation of the state to be achieved or maintained). As with the simple reflexive behaviours described above, this kind of architecture supports conditionals, looping etc., though in this case we can loop a fixed number of times without having to record anything in the external environment, but the sets of rules can be more naturally described as simple control programs rather than collections of simple reflexes.

Although reactive behaviour is essentially data-driven, it may be useful to conceive of it as being goal-directed. We distinguish between two types of goals: implicit and explicit. Implicit goals are not subject to deliberation. We can view a reactive behaviour which is triggered by, e.g., an approaching object (and hence the possibility of a collision) as a response to an implicit goal of avoiding collisions. For purely data-driven reactive behaviours, this way of looking at things adds nothing, but it provides a uniform interface to behaviours which can either be triggered by events in the world or under the control of the agent. In particular, it gives us a way of indexing reactive behaviours by their effects, so that we can find behaviours appropriate to a deliberative goal.

An *explicit* goal is generated whenever the reactive layer doesn't know what to do. We assume that the agent has some sort of control program or reactive behaviour for all classes of events which are of interest to the agent. Percepts or representations of the world are matched against the agent's reactive behaviours, and some or all of the behaviours which match are invoked. Any 'percepts' or

[1] Specifically it is not intended to be a general representational capability, and cannot handle beliefs about things which are not present to the agents senses (unless they were previously), or indeed about many of the things which are. However it could be viewed as the first step towards such a representational capability.

[2] An arrangement in which 'world-to-representation' and 'representation-to-novelty' rules is replaced by a single rule with conditions which match both percepts and internal representations would of course also work, but the notion of a rule which only responds to and generates internal changes in the agent is a key step, since it forms the basis of all derived representations, and of representations which refer to other aspects of the agent's internal state.

sense data which don't trigger some sort of behaviour are simply ignored by the agent. This is perhaps more credible if we assume an agent with several levels of perceptual processing, where a failure to produce any sort of perceptual classification for a for low-level percept, e.g. hearing an unfamiliar noise, may give rise to an explicit goal, e.g. to investigate the source of the noise. If a percept matches a behaviour but the behaviour cannot be executed (perhaps because a precondition is not satisfied), or the behaviour fails during execution and the reactive layer is not able to recover, then an explicit goal is generated by some error handling mechanism in the behaviour. Similarly, if a percept matches several behaviours, this could also give rise to a deliberative goal. We may want to assume some sort of heuristic partial order over behaviours, or many such orderings (e.g. effectiveness, reliability, speed etc.), with different orderings being used in different situations. Only in the case in which there is no clear preferred behaviour is an explicit goal generated. The result is an architecture which is basically reactive, in which deliberation is used as a last resort. The agent only engages in explicit deliberation about what to do next when a reactive behaviour 'fails' in some way. What happens when a new explicit goal is generated is discussed in more detail in section 5.

3 The deliberative layer

An explicit goal is one which involves (explicit) deliberation, for example the explicit generation and evaluation of alternative courses of action by the agent, or a decision to perform some action. Such explicit goals can be either *intrinsic* or *derived*. An intrinsic goal is one which is not a subgoal of an already intended end. A derived goals is a subgoal generated in response to an existing intrinsic or derived goal.[3] Some steps in the deliberation may take the form of basic actions (possibly expressed as implicit goals, see above), which simply trigger some reactive behaviour; for example, adding two single digit integers, or comparing two alternative actions to decide which is preferable.

Beaudoin and Sloman (Beaudoin and Sloman, 1993) identify ten fields associated with an (explicit) goal including: a possible state of affairs p; a propositional attitude towards p; a value representing the agent's current beliefs about p; an importance value relating to the benefits of achieving the goal or the costs of not achieving the goal; the urgency of the goal; a plan or set of plans for achieving the goal; a commitment status such as 'adopted', 'rejected' or 'undecided'; and management or scheduling information in the form of condition action pairs, determining, for example, when execution of the plan or further deliberation should begin or be resumed if it is currently suspended.[4] The urgency and importance of a goal can be thought of as modifiers of the basic propositional attitude ranging along two (possibly orthogonal) dimensions. The degree of urgency or importance of a goal is relative to that of the other goals the agent is currently trying to achieve and therefore can't be determined *a priori*. Goals also have an insistence value which is a heuristic estimate of the goal's likely urgency and importance (and possibly other things), which can be defined operationally as the ability of the goal to penetrate the attention filter. In what follows we will focus on the commitment status, management information and insistence of a goal.

The basic unit at the deliberative level is the *task*. A task is a declarative representation of a sequence of actions to achieve a goal. Deliberative tasks organise or synchronise reactive behaviours in pursuit of an explicit goal, and may involve complex internal organisation and the explicit anticipation of possible futures. In contrast to reactive behaviours they are goal rather than data-driven, and can involve commitment to future action.

A task has several components:

- the goal the task is trying to achieve, including any constraints on how the goal can be achieved;
- when the task has been scheduled, i.e. when we intend to start work on the task; and
- any preconditions that must be true before we can perform the task.

A task can be thought of as stack consisting of an initial intrinsic goal and the unachieved derived subgoals generated in attempting to achieve the original goal. The top of the stack is the current subgoal and the preconditions are the conjunction of the preconditions for the pending actions. Tasks are executed by a deliberative 'interpreter' which is itself simply a collection of appropriately organised reactive behaviours responsible for pattern matching, manipulation of the goal stack, updating working memory etc.

The scheduling conditions of a task define a partial order over tasks which tells the agent which tasks it should be working on at any one time. As used here, 'scheduling' is to be understood loosely as any absolute or relative temporal ordering over tasks, e.g., "I'll do it on Friday", or "I'll do it after I have finished debugging this function". In this model, Bratmans's (Bratman, 1987) notion of a 'committed intention' is equivalent to a task that has been scheduled. A task will not usually be considered for execution until all its scheduling conditions evaluate to true. The rationale for this is that if we have decided that we won't do something until next week, there is no point in continually checking the preconditions for the task to see if it could be executed next. Presumably the task was

[3] Georgeff and Lansky (Georgeff and Lansky, 1986) call these *operational* goals.

[4] Note that the use of some terms, e.g., 'adoption', differs from that in (Beaudoin and Sloman, 1993).

scheduled next week for a reason: for example, it may be that we can't do it sooner, or that we anticipate that we will have more than enough to do in the meantime.

When the scheduling conditions for a task evaluate to true, the task can be considered for execution. A task is executable if its preconditions evaluate to true. By default, new tasks are created with no scheduling information or preconditions (unless the task is a subtask of an existing task, see above). New preconditions are added whenever the agent must wait for some condition to be true in the world before the task can continue, for example for an action to complete (such as travelling to a particular location) or for a resource to become available. The preconditions can be any state in the world or the agent and will usually be dependent on the level of the task. For example, a repair task may be suspended awaiting the delivery of a replacement part, or for a supplier to confirm availability or price of a part, whereas a scheduling (management) task, might be suspended awaiting an estimate of how long the task will take to execute, or whether a particular way of achieving a goal is feasible. If one or more of the preconditions of a task evaluate to false, the task is suspended until the precondition becomes true.

Processing at the deliberative and reactive layers proceeds concurrently. While, the current deliberative task is the focus of the agent's attention, deliberation associated with the current task proceeds in parallel with the execution of actions associated with any suspended tasks by the reactive layer. For example, we may be walking down the corridor in pursuit of an explicit deliberative goal (to get some coffee) while thinking about something else.

4 Managing deliberation

Periodically, the agent must consider whether it should continue with its current deliberative task or switch to another. This is the process loosely referred to as 'scheduling' in the previous section.

The scheduler is an independent process which runs in parallel with the deliberative interpreter, and which continually monitors its progress. When a new goal penetrates the filter, or if the scheduler detects that the current task is not making progress, or the scheduling condition for a higher priority task has just become true, the scheduler arranges for the deliberative interpreter to stop executing the current task and start executing the new task. If making such a decision itself requires deliberation, the scheduler creates a new deliberative scheduling task, and causes the deliberative interpreter to switch to this task.[5]

Scheduling can be non pre-emptive if we can assume an upper bound on the time necessary to execute a basic action, or that the initiation of a durative action causes the task to be suspended while waiting for the action to complete (e.g., (Georgeff and Lansky, 1986)). Alternatively, if actions can take a substantial amount of time to complete or we cannot suspend the current task while the action is executing (for example, if it is not easy to determine a success condition for the action to use as a precondition), then it may be more appropriate to use pre-emptive scheduling (which may result in any work for partially complete subgoals/basic actions being lost).

One factor which is important in determining what to do next is how much progress has been made towards the current goal. A plan may be of the form "do X" (e.g., try and jiggle a part into a socket, or try and find a plan) or of the form "do X for a while, and if that doesn't work, try Y". In the former case, we have to rely on general monitoring by the scheduler to discover that we are not making progress. If a behaviour has an 'outcome' related to the implicit or explicit goal the behaviour is trying to achieve (rather than simply a 'success' or 'failure' flag), then the scheduler can look at the outcome and decide whether to terminate the behaviour, allow it to continue, or suspend it to allow another task to run. One way the current outcome of a behaviour can be determined is to compute the (relevant) change in the world since the behaviour started execution, or to compare the current state of the world with the goal. Information about the current outcome could be used by an explicit monitor for the plan, or could form some sort of annotation on the task to tell the scheduler how long to persist with X before switching to Y (all other things being equal). Like other behaviours, progress monitoring is partially reactive and partially deliberative. Routine monitoring is handled reactively, with goals being generated if the reactive progress monitoring behaviours can't determine if progress is being made.

When the scheduler decides that X is not achieving its goal, we must either replan (if this contingency was not anticipated) or switch to Y (if it was). For example, we could arrange for the scheduler to send some sort of 'failure' control signal to the current task, causing it to backtrack and select another behaviour. In the worst case, this would lead to the failure of the top-level goal for the task, which could trigger replanning. This has the advantage of localising plan repair within the task or plan. However it implies a task representation with an explicit notion of 'failure'. Alternatively, we could exercise control directly at the meta-level. A plan is not a behaviour; it is a declarative structure, the major steps in which are explicitly executed. We could use this representation to reason explicitly at the meta-level (perhaps using task-specific knowledge) about where to backtrack to and which other behaviours might achieve the goal.

We assume that most of the decisions about which task to switch to next can be be made quickly by the data-driven or reactive parts of the scheduler responding to automatically generated meta-level descriptions of the current state of the system's deliberative tasks, and we only use deliberation to determine what to do next as a last re-

[5]The alternative approach of giving the the scheduler its own deliberative interpreter doesn't seem to have the right phenomenology, since we want such deliberative scheduling tasks to interrupt the current deliberative task, whatever it is.

sort.[6] The default scheduling policy may be to stick with the current task unless the scheduling conditions or preconditions of another task have just become true, or we are not making progress with what we are doing, for example if the task is taking longer than estimated, when it may be necessary to generate an explicit management goal. Initially, any unusual situation may give rise to a management goal, but with experience, the agent may develop reactive behaviours for the more common cases, perhaps as a result of a process similar to chunking in SOAR (Laird et al., 1987).

5 New intrinsic goals

As described in the preceding sections, any events which can't be handled by the reactive behaviours of the agent give rise to new intrinsic goals. In this section, we discuss filtering of new intrinsic goals and the process of goal adoption, and argue that the decision whether to adopt a goal can most usefully be viewed as part of the scheduling process.

When a goal is generated by the reactive layer, it must first pass the attention filter to be eligible for consideration by the mechanisms at the deliberative layer. The purpose of the attention filter is to avoid interrupting the current task unless it is likely that the new goal is both urgent and important in the current context. In general, it is impossible to determine the urgency or importance of a goal, and hence if we should consider the goal further, without expending (possibly considerable) deliberative resources. By assumption, such deliberative resources are not available to the filter mechanism, and passing the problem to the deliberative layer would defeat the point of having the filter in the first place. Instead, the attention filter uses simple heuristics based on the goal's insistence and the current filter threshold to determine if the goal should receive further consideration. The attention filter will therefore be prone to 'errors', passing goals which on further consideration are not worth adopting, and failing to pass goals which it would have been beneficial for the agent to have adopted.

Whether an agent should *in fact* attempt to achieve a goal may depend on many factors, and may require considerable deliberation, for example deciding whether one should agree to organise a workshop at a conference or write a paper about a particular piece of work. Consideration of how, when and ultimately whether to intend some state X may be spread over a protracted period, and it is difficult to justify the choice of a particular point as *the* point at which a goal to achieve X is adopted. Rather there is a wide range of intermediate states each of which may affect the way the agent responds to the putative goal and its other tasks in different ways. For example, we may not have decided how urgent or important a goal is, or consideration of the goal may be suspended pending information necessary to make a decision whether to pursue it further (e.g., the information about the cost of travelling to X may be instrumental in deciding whether we will go there), or the goal may have been provisionally adopted but only more or less loosely scheduled, e.g., "I'll do it if I have time", to "I'll do it next week (sometime)", to "I'll do it on Wednesday at 4pm" to "I'll do it next" to "I'll do it now". Moreover, while Bratman (1987) is correct in that we don't continually reconsider our commitment to our existing intentions and that such commitments provide an essential framework for planning, intermediate stages prior to 'full' commitment can still be useful for scheduling. For example, I may decide not to accept a dinner invitation next week because I believe that I have a lot of things tentatively scheduled for next week, and if pressed to make a decision at short notice may conclude that I am unlikely to be able to fit everything in. Given more time to decide, I may have been able to schedule all my current intentions and still accept the invitation. In what follows, we assume that new goals are automatically 'adopted', at least provisionally, after passing the attention filter, and that a new task is created for them.

Once it has been adopted, the processing of a new intrinsic goal is similar to the scheduling of existing tasks. The first time the scheduler runs after the adoption of a new intrinsic goal, there is a new task with no information associated with it, for example, the task goal will typically have no urgency or importance fields. This lack of information makes it difficult to use heuristics to decide whether to switch to the new task (and more generally what to do with it, e.g., whether it should be deleted), unless either the new task or the current task is extremely urgent and/or important, causing all other tasks to be ignored. For example, the sound of a fire alarm and the resulting goal to leave the building immediately may be sufficiently unambiguous that it takes precedence over all other deliberative tasks. Usually however, the failure of the heuristic scheduling behaviours result in the generation of a new explicit management goal to find out more about the task to determine if we should switch to it. This may involve working out how the task can be done, how urgent and important it is (relative to the other tasks the agent currently intends) and so on, and ultimately results in a heuristic or deliberative scheduling decision.

At first sight, it may seem that new management goals must be handled in a slightly different way. If we try to decide what to do next and can't, we generate a goal to think about it. This goal is like a new intrinsic goal in that it is generated reactively; moreover it is not a derived goal of any of the existing tasks. However, it seems unreasonable that a management goal should have to pass the attention filter, as if it *failed* to pass the filter (with the result that we fail to attend to the goal) we still wouldn't know what to do next. Similarly, if it passes the filter but isn't adopted or we don't switch to processing it, we are

[6] If several scheduling rules apply, we could pick one at random or allow them to 'vote' for which task to run next. If they disagree, then this might be another reason for generating a meta-management goal.

still stuck. Management goals must therefore have special status within the architecture.

However, the context in which a management goal is generated is usually such as to ensure that it will be processed next anyway, without having to take special measures. If it is not clear what to do next, the filter threshold is presumably not very high, since the presence of an urgent and/or important task (which could easily be scheduled reactively) implies a high filter threshold. If management goals generally have a high insistence value, there is at least a good chance that a management goal will be selected as the current deliberative task at the next iteration of the scheduler. This approach has the advantage of allowing preemption of management goals by new, urgent 'ordinary' goals—the context changes and it suddenly becomes clear what we should do or think about next—and it may make it easier to explain certain types of perturbances where meta-management fails.

6 Summary

In this paper, we have sketched a design for the reactive-deliberative interface in an Attention Filter Penetration architecture. In doing so, we have tried to explain how new explicit goals are processed within the architecture and how it is that some goals are capable of redirecting the attention of the deliberative system.

Working within the framework outlined in (Sloman, 2000), we have sketched the progress of a new explicit goal from its initial generation in the reactive layer in response to a reactive failure, through the attention filter and into the deliberative layer, and briefly described how the new goal interacts with existing goals at the deliberative layer. We have focussed on the role of the scheduling mechanism which decides which goal to work on next, and argued that scheduling, like the other behaviours of the agent, is partly reactive and partly deliberative. Reactive scheduling failures give rise to management goals. At first sight, it may seem that such goals must have a special place within the architecture. However, we have argued that the context in which a management goal is generated is usually such as to ensure that it will be processed next anyway, without having to take special measures. This allows preemption of management goals by new, urgent 'ordinary' goals—the context changes and it suddenly becomes clear what we should do or think about next—and it may make it easier to explain certain types of perturbances where meta-management fails.

We have left many important questions unanswered. Further work is required to clarify (among other things) the nature of the deliberative-reactive interface: how the deliberative layer can invoke reactive behaviours and modulate their execution. Humans seem to be able to execute and modify a wide range of reactive behaviours under deliberative control, descending recursively through the component behaviours of a compound behaviour to obtain the required degree of control. For example, we may execute some steps in a plan essentially unconsciously, while closely controlling other steps, or parts of steps. More work is also required to elaborate the detailed behaviour of the attention filter and scheduler. This is the subject of current research.

Acknowledgements

Aaron Sloman, Luc Beaudoin and Ian Wright read an earlier version of this paper and made many helpful comments.

References

Luc P. Beaudoin. *Goal processing in autonomous agents*. PhD thesis, School of Computer Science, The University of Birmingham, 1994.

Luc P. Beaudoin and Aaron Sloman. A study of motive processing and attention. In A. Sloman, D. Hogg, G. Humphreys, D. Partridge, and A. Ramsay, editors, *Prospects for Artificial Intelligence*, pages 229–238. IOS Press, Amsterdam, 1993.

Michael E. Bratman. *Intention, Plans, and Practical Reason*. Harvard University Press, 1987.

Nico H. Frijda. *The emotions*. Cambridge University Press, Cambridge, 1986.

M. P. Georgeff and A. L. Lansky. A system for reasoning in dynamic domains: Fault diagnosis on the space shuttle. Technical Report Technical Note 375, Artificial Intelligence Center, SRI International, Menlo Park, California, 1986.

J. E. Laird, A. Newell, and P. S. Rosenbloom. SOAR: An architecture for general intelligence. *Artificial Intelligence*, 33:1–64, 1987.

A. Sloman. Explorations in design space. In A.G. Cohn, editor, *Proceedings 11th European Conference on AI, Amsterdam, August 1994*, pages 578–582, Chichester, 1994. John Wiley.

A. Sloman. What sort of control system is able to have a personality. In Robert Trappl and Paolo Petta, editors, *Creating Personalities for Synthetic Actors: Towards Autonomous Personality Agents*, pages 166–208. Springer (Lecture Notes in AI), Berlin, 1997.

A. Sloman. Damasio, Descartes, alarms and meta-management. In *Proceedings International Conference on Systems, Man, and Cybernetics (SMC98)*, pages 2652–7. IEEE, 1998.

A. Sloman and M. Croucher. Why robots will have emotions. In *Proceedings of the 7th International Joint Conference on AI*, pages 197–202, Vancouver, 1981.

A. Sloman and R. Poli. Sim_agent: A toolkit for exploring agent designs. In Mike Wooldridge, Joerg Mueller, and Milind Tambe, editors, *Intelligent Agents Vol II (ATAL-95)*, pages 392–407. Springer-Verlag, 1996.

Aaron Sloman. Architectural requirements for human-like agents both natural and artificial. (what sorts of machines can love?). In Kerstin Dautenhahn, editor, *Human Cognition And Social Agent Technology*, Advances in Consciousness Research, pages 163–195. John Benjamins, Amsterdam, 2000.

Aaron Sloman and Brian Logan. Building cognitively rich agents using the Sim_agent toolkit. *Communications of the Association of Computing Machinery*, 42 (3):71–77, March 1999.

Ian P. Wright. *Emotional agents*. PhD thesis, School of Computer Science, The University of Birmingham, 1997.